WEB APP RECON TOOLS

Laboratory Exercise 6.1 – Web App Recon Tools

1. Overview

For this lesson, students will use the latest Cyber Range: Cyber Basics Environment to install several tools to complete web application reconnaissance in later lessons.

2. Resources required

This exercise requires the latest Cyber Basics Environment running in the Cyber Range.

3. Initial Setup

For this exercise, you will log in to your Cyber Range account and select the latest Cyber Basics Environment, then click "start" to start your environment and "join" to get to your Linux desktop login. Log in using these credentials:

Username: **`student`**
Password: **`student`**

4. Tasks

Task 1: Installing GO Language

Many tools use go language for installation to make the process easier. This will also help beginners using the scripting techniques discussed in later lessons. Open a terminal and switch to the root user. Complete the following:

- In a root terminal, type **`apt update`** and press enter.
- When the update finishes, type **`apt install golang`** and press enter. When prompted, type **`Y`** and hit enter.

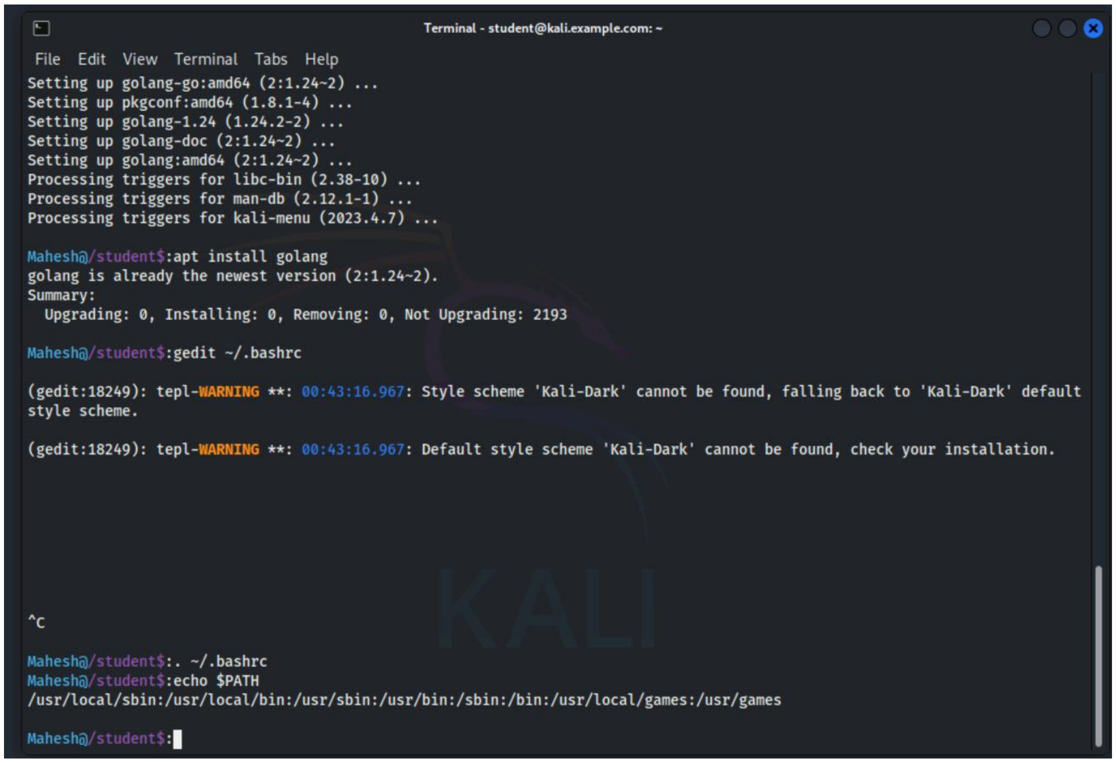
For bash to know where the GO language folder is located, we have to export the path. For this, we will use **gedit** and make some changes to the .bashrc file.

- Type **gedit ~/.bashrc** and press enter.
- Add the following to the bottom of the file:

```
export GOPATH=$HOME/go
export PATH=$PATH:$GOPATH/bin
```

- Save the changes, exit gedit, and return to the terminal.
- Refresh the terminal profile by typing:
  `. ~/.bashrc` and pressing enter.

- To test the path, type `echo $PATH` and press enter. You should see go/bin in the directory path.



Terminal - student@kali.example.com: ~
File Edit View Terminal Tabs Help

```
Setting up golang-go:amd64 (2:1.24~2) ...
Setting up pkgconf:amd64 (1.8.1-4) ...
Setting up golang-1.24 (1.24.2-2) ...
Setting up golang-doc (2:1.24~2) ...
Setting up golang:amd64 (2:1.24~2) ...
Processing triggers for libc-bin (2.38-10) ...
Processing triggers for man-db (2.12.1-1) ...
Processing triggers for kali-menu (2023.4.7) ...

Mahesh@/student$:apt install golang
golang is already the newest version (2:1.24~2).
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 2193

Mahesh@/student$:gedit ~/.bashrc

(gedit:18249): tepl-WARNING **: 00:43:16.967: Style scheme 'Kali-Dark' cannot be found, falling back to 'Kali-Dark' default
style scheme.

(gedit:18249): tepl-WARNING **: 00:43:16.967: Default style scheme 'Kali-Dark' cannot be found, check your installation.




^C

Mahesh@/student$:. ~/.bashrc
Mahesh@/student$:echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/games:/usr/games

Mahesh@/student$:
```

The GO language installation is complete and can be used to install the needed tools.

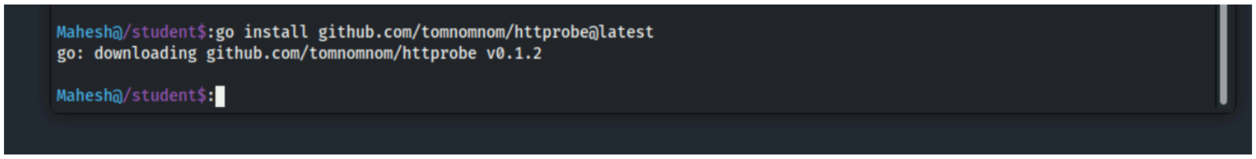
Task 2: Installing HTTProbe, and Asset Finder

As discussed in the PowerPoint for this lesson, Httprobe is an extremely fast way of checking if ports 80 and 443 are open on a domain/subdomain. By creating a list of subdomains with Amass, we can run that list through HTTProbe to determine what systems are public-facing.

- Install Httprobe using a root terminal and typing:

  `go install github.com/tomnomnom/httprobe@latest` and press enter.

If the terminal returns to the prompt without errors, you have successfully installed HTTProbe.



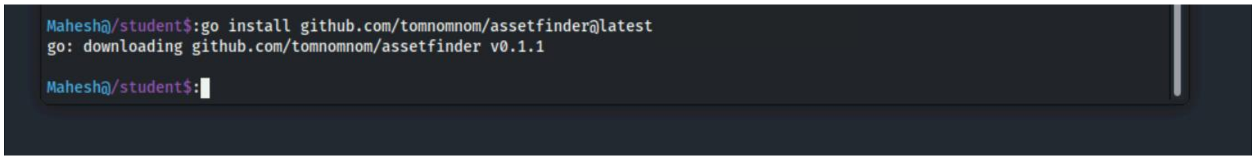
```
Mahesh@/student$:go install github.com/tomnomnom/httprobe@latest
go: downloading github.com/tomnomnom/httprobe v0.1.2

Mahesh@/student$:
```

Asset Finder is much faster than Amass because it uses database parsing to discover domains and subdomains. The program is a script that looks at several sources for domain information based on the input. In other words, if someone else has stored information on a domain this script will grab that information. This is faster but less accurate.

- Install Asset Finder using a root terminal and typing:

  `go install github.com/tomnomnom/assetfinder@latest` and press enter.

If the terminal returns to the prompt without errors, you have successfully installed Asset Finder.



```
Mahesh@/student$:go install github.com/tomnomnom/assetfinder@latest
go: downloading github.com/tomnomnom/assetfinder v0.1.1

Mahesh@/student$:
```

Task 3: Installing Dirsearch

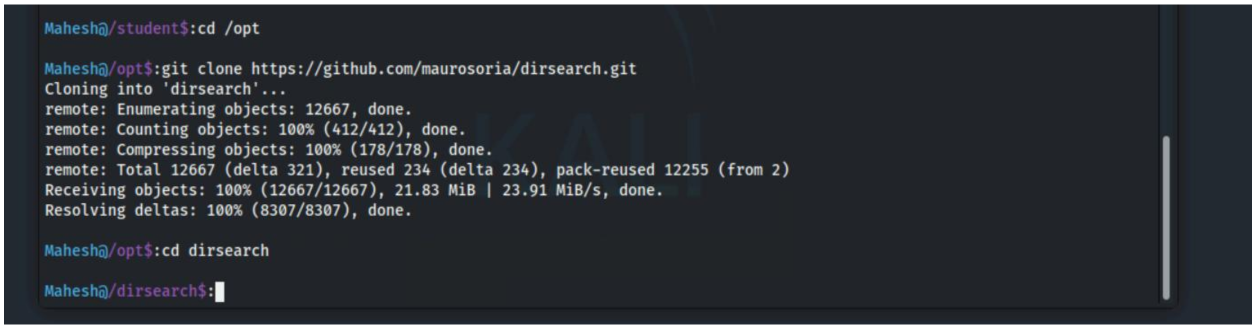
Dirsearch is a fast directory brute-forcing tool. As we know all websites have directories that show their content. Not all content can be viewed. However, much of the content has standard or default naming conventions. This allows a wordlist to be used to brute force those locations.

- Install Dirsearch by using a root terminal by typing:

  `cd /opt` and press enter. Then type:

  `git clone https://github.com/maurosoria/dirsearch.git` and press enter.

- To verify the Dirsearch installation, type **`cd dirsearch`** and press enter.

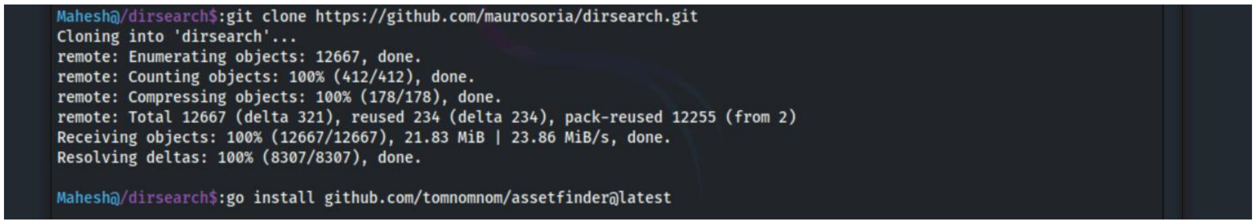


```
Mahesh@/student$:cd /opt

Mahesh@/opt$:git clone https://github.com/maurosoria/dirsearch.git
Cloning into 'dirsearch'...
remote: Enumerating objects: 12667, done.
remote: Counting objects: 100% (412/412), done.
remote: Compressing objects: 100% (178/178), done.
remote: Total 12667 (delta 321), reused 234 (delta 234), pack-reused 12255 (from 2)
Receiving objects: 100% (12667/12667), 21.83 MiB | 23.91 MiB/s, done.
Resolving deltas: 100% (8307/8307), done.

Mahesh@/opt$:cd dirsearch

Mahesh@/dirsearch$:
```

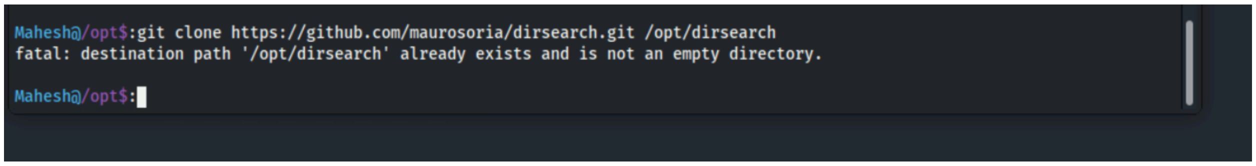
- Next type **`./dirsearch.py`** and press enter. If you get an error **"URL target is missing, try using -u <url>,"** Dirsearch has successfully installed.

- You may have to Type `Y` and press enter to allow dirsearch to automatically install dependencies.

- If you receive the **"Failed to install dirsearch dependencies, try doing it manually"** error then type the following `pip3 install -r requirements.txt` and then Enter.



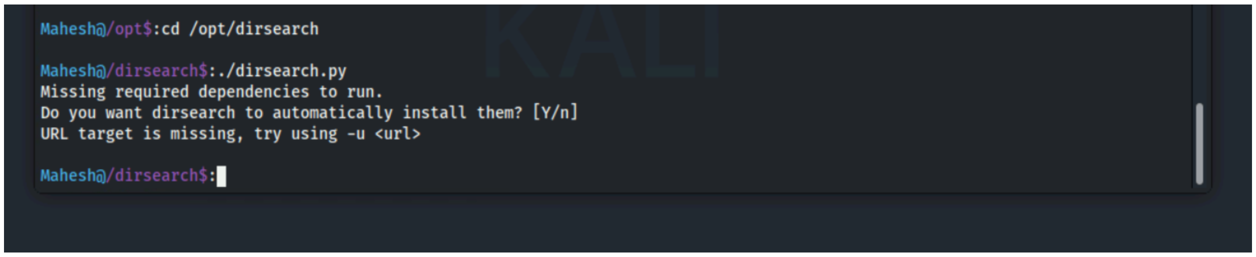
```
Mahesh@/dirsearch$:git clone https://github.com/maurosoria/dirsearch.git
Cloning into 'dirsearch'...
remote: Enumerating objects: 12667, done.
remote: Counting objects: 100% (412/412), done.
remote: Compressing objects: 100% (178/178), done.
remote: Total 12667 (delta 321), reused 234 (delta 234), pack-reused 12255 (from 2)
Receiving objects: 100% (12667/12667), 21.83 MiB | 23.86 MiB/s, done.
Resolving deltas: 100% (8307/8307), done.

Mahesh@/dirsearch$:go install github.com/tomnomnom/assetfinder@latest
```



```
Mahesh@/opt$:git clone https://github.com/maurosoria/dirsearch.git /opt/dirsearch
fatal: destination path '/opt/dirsearch' already exists and is not an empty directory.

Mahesh@/opt$:
```

- **REDO THIS STEP:** Type **./dirsearch.py** and press enter. If you get an error **"URL target is missing, try using -u <url>,"** Dirsearch has successfully installed.

- This time, select `n` and press enter when prompted for dirsearch to automatically install dependencies.



```
Mahesh@/opt$:cd /opt/dirsearch

Mahesh@/dirsearch$:./dirsearch.py
Missing required dependencies to run.
Do you want dirsearch to automatically install them? [Y/n]
URL target is missing, try using -u <url>

Mahesh@/dirsearch$:
```

Task 4: Creating an alias for Dirsearch

To create an alias for Dirsearch, type the following in a root terminal:

> `aliases` and press enter. (Recall, that this is an alias we created in the last module.) The .bash_aliases file will open in gedit.

At the bottom of the document, add the following:

```
alias dirsearch="cd /opt/dirsearch ; ./dirsearch.py -x502,503,500 -u"
```

Then save and exit the document.

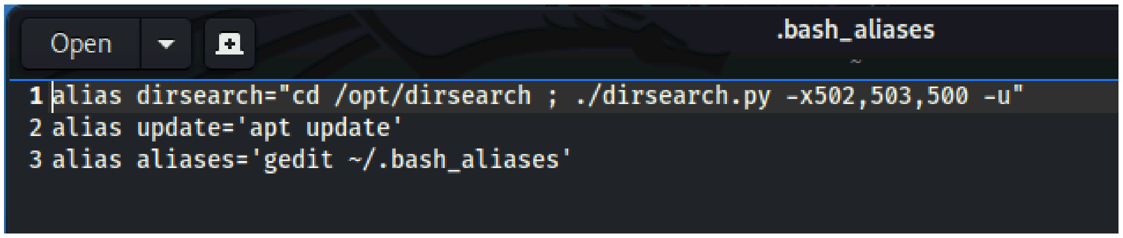
Refresh the terminal profile by typing:

`. ~/.bash_aliases` and pressing enter. (**NOTE**: There is a space between the `.` and `~/`)

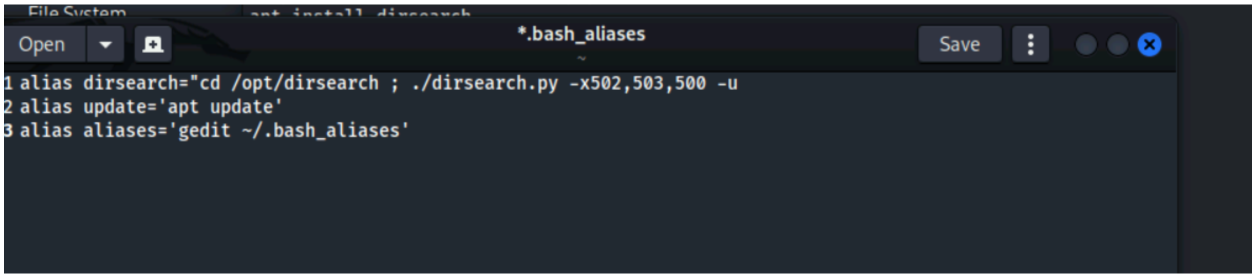
In the terminal, type **`dirsearch`** and press enter. If you get an error **"-u option requires 1 argument,"** then the alias was successful.

Explanation of the alias:

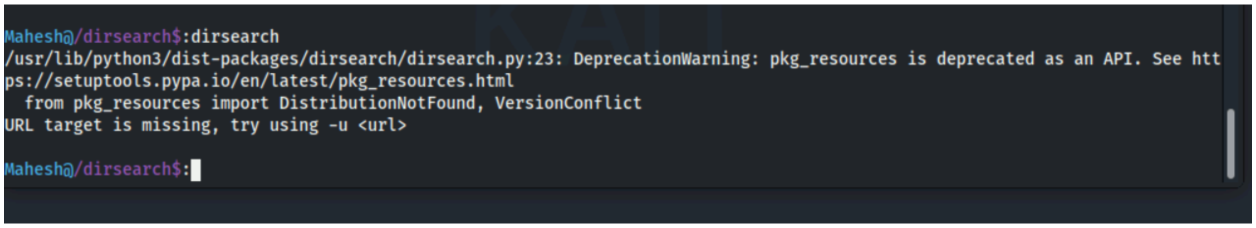
`-x`	excludes the specified page errors from the results.
`;`	joins two commands together.
`-u`	added so we do not have to type `–u` every time we run the command.



```
alias dirsearch="cd /opt/dirsearch ; ./dirsearch.py -x502,503,500 -u"
alias update='apt update'
alias aliases='gedit ~/.bash_aliases'
```



```
alias dirsearch="cd /opt/dirsearch ; ./dirsearch.py -x502,503,500 -u
alias update='apt update'
alias aliases='gedit ~/.bash_aliases'
```



```
Mahesh@/dirsearch$:dirsearch
/usr/lib/python3/dist-packages/dirsearch/dirsearch.py:23: DeprecationWarning: pkg_resources is deprecated as an API. See htt
ps://setuptools.pypa.io/en/latest/pkg_resources.html
  from pkg_resources import DistributionNotFound, VersionConflict
URL target is missing, try using -u <url>

Mahesh@/dirsearch$:
```

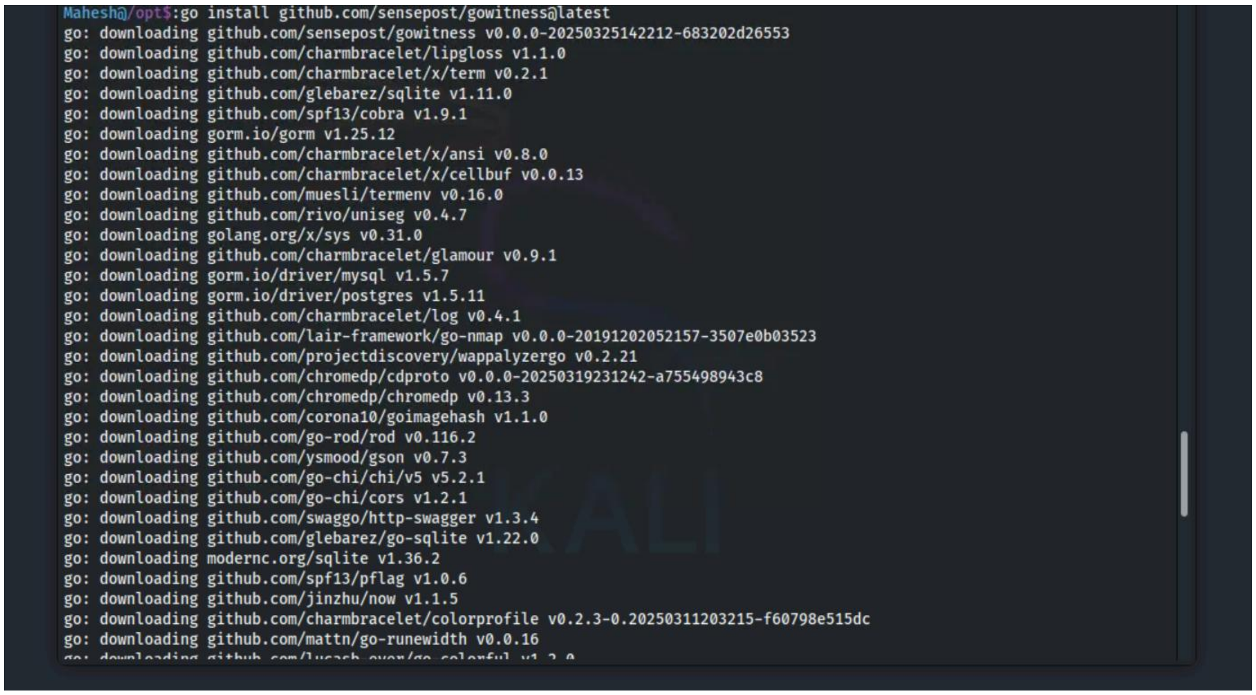
Task 5: Installing GOwitness

GOwitness will take screenshots of all live websites from a list of domains/subdomains. These images can then be viewed to determine if there are interesting sites to test for vulnerabilities.

Install GOwitness by using a root terminal by typing:

`go install github.com/sensepost/gowitness@latest` and press enter.

If the terminal returns to the prompt without errors, you have successfully installed GOwitness.



```
Mahesh@/opt$:go install github.com/sensepost/gowitness@latest
go: downloading github.com/sensepost/gowitness v0.0.0-20250325142212-683202d26553
go: downloading github.com/charmbracelet/lipgloss v1.1.0
go: downloading github.com/charmbracelet/x/term v0.2.1
go: downloading github.com/glebarez/sqlite v1.11.0
go: downloading github.com/spf13/cobra v1.9.1
go: downloading gorm.io/gorm v1.25.12
go: downloading github.com/charmbracelet/x/ansi v0.8.0
go: downloading github.com/charmbracelet/x/cellbuf v0.0.13
go: downloading github.com/muesli/termenv v0.16.0
go: downloading github.com/rivo/uniseg v0.4.7
go: downloading golang.org/x/sys v0.31.0
go: downloading github.com/charmbracelet/glamour v0.9.1
go: downloading gorm.io/driver/mysql v1.5.7
go: downloading gorm.io/driver/postgres v1.5.11
go: downloading github.com/charmbracelet/log v0.4.1
go: downloading github.com/lair-framework/go-nmap v0.0.0-20191202052157-3507e0b03523
go: downloading github.com/projectdiscovery/wappalyzergo v0.2.21
go: downloading github.com/chromedp/cdproto v0.0.0-20250319231242-a755498943c8
go: downloading github.com/chromedp/chromedp v0.13.3
go: downloading github.com/corona10/goimagehash v1.1.0
go: downloading github.com/go-rod/rod v0.116.2
go: downloading github.com/ysmood/gson v0.7.3
go: downloading github.com/go-chi/chi/v5 v5.2.1
go: downloading github.com/go-chi/cors v1.2.1
go: downloading github.com/swaggo/http-swagger v1.3.4
go: downloading github.com/glebarez/go-sqlite v1.22.0
go: downloading modernc.org/sqlite v1.36.2
go: downloading github.com/spf13/pflag v1.0.6
go: downloading github.com/jinzhu/now v1.1.5
go: downloading github.com/charmbracelet/colorprofile v0.2.3-0.20250311203215-f60798e515dc
go: downloading github.com/mattn/go-runewidth v0.0.16
go: downloading github.com/lucasb-eyer/go-colorful v1.2.0
```

Task 6: Installing Knockpy

Knockpy is a subdomain brute forcing tool that uses a wordlist. This makes it great for internal and external subdomain discovery. It also can check for a DNS zone transfer and bypass wildcard DNS records. Knockpy needs Google Chrome to work properly so we will install this as well.

To install Knockpy complete the following:

In a root terminal, type:

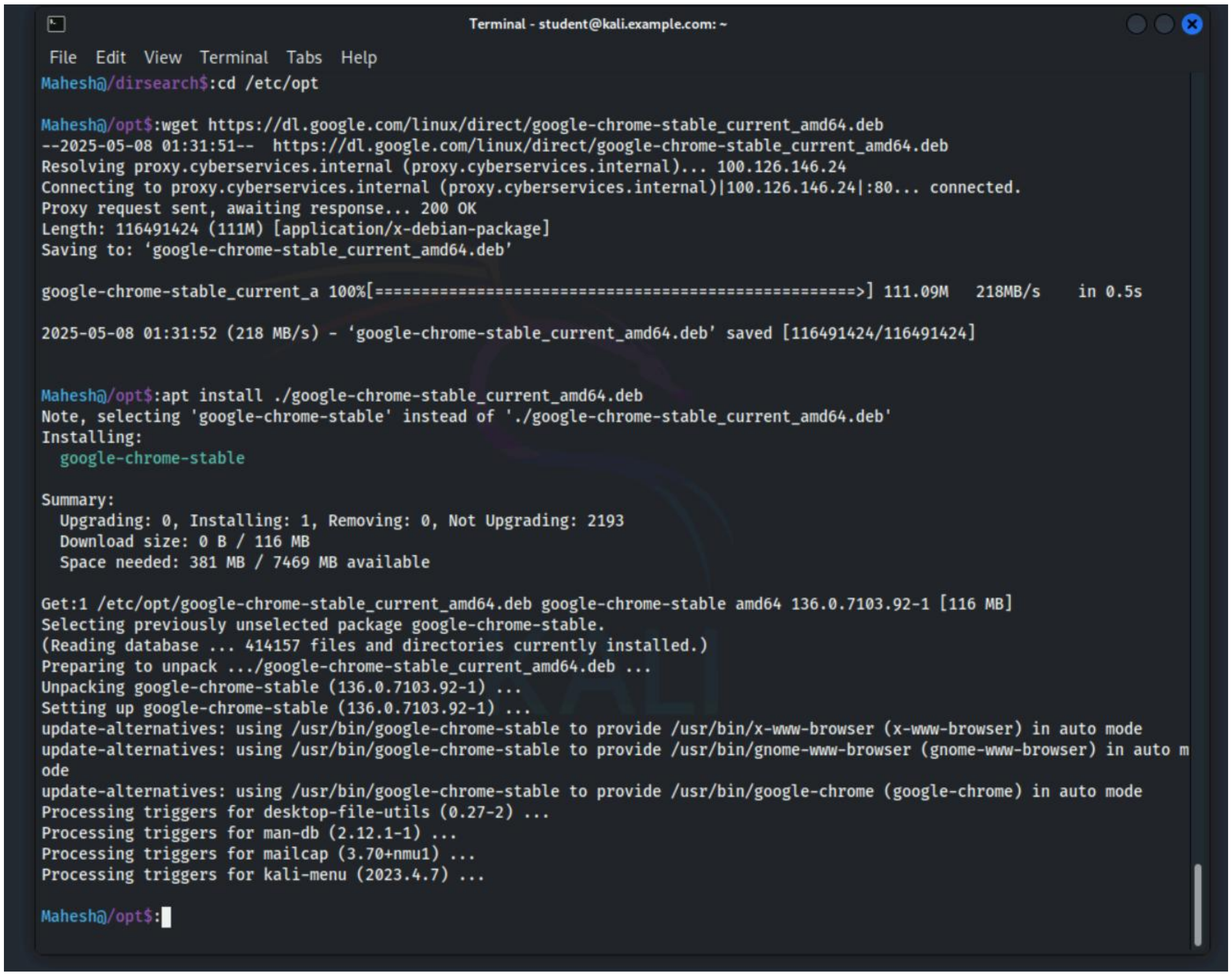
`cd /etc/opt` and press enter.

- Type

`wget https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb` and press enter.

- Type

  `apt install ./google-chrome-stable_current_amd64.deb` and press enter. If prompted, type **`y`** and press enter.



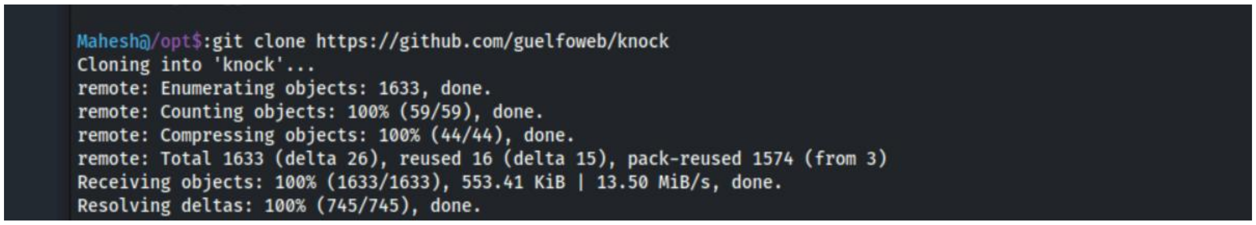
Terminal - student@kali.example.com: ~
File Edit View Terminal Tabs Help

```
Mahesh@/dirsearch$:cd /etc/opt

Mahesh@/opt$:wget https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb
--2025-05-08 01:31:51--  https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb
Resolving proxy.cyberservices.internal (proxy.cyberservices.internal)... 100.126.146.24
Connecting to proxy.cyberservices.internal (proxy.cyberservices.internal)|100.126.146.24|:80... connected.
Proxy request sent, awaiting response... 200 OK
Length: 116491424 (111M) [application/x-debian-package]
Saving to: 'google-chrome-stable_current_amd64.deb'

google-chrome-stable_current_a 100%[=================================================>] 111.09M   218MB/s    in 0.5s

2025-05-08 01:31:52 (218 MB/s) - 'google-chrome-stable_current_amd64.deb' saved [116491424/116491424]


Mahesh@/opt$:apt install ./google-chrome-stable_current_amd64.deb
Note, selecting 'google-chrome-stable' instead of './google-chrome-stable_current_amd64.deb'
Installing:
  google-chrome-stable

Summary:
  Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 2193
  Download size: 0 B / 116 MB
  Space needed: 381 MB / 7469 MB available

Get:1 /etc/opt/google-chrome-stable_current_amd64.deb google-chrome-stable amd64 136.0.7103.92-1 [116 MB]
Selecting previously unselected package google-chrome-stable.
(Reading database ... 414157 files and directories currently installed.)
Preparing to unpack .../google-chrome-stable_current_amd64.deb ...
Unpacking google-chrome-stable (136.0.7103.92-1) ...
Setting up google-chrome-stable (136.0.7103.92-1) ...
update-alternatives: using /usr/bin/google-chrome-stable to provide /usr/bin/x-www-browser (x-www-browser) in auto mode
update-alternatives: using /usr/bin/google-chrome-stable to provide /usr/bin/gnome-www-browser (gnome-www-browser) in auto m
ode
update-alternatives: using /usr/bin/google-chrome-stable to provide /usr/bin/google-chrome (google-chrome) in auto mode
Processing triggers for desktop-file-utils (0.27-2) ...
Processing triggers for man-db (2.12.1-1) ...
Processing triggers for mailcap (3.70+nmu1) ...
Processing triggers for kali-menu (2023.4.7) ...

Mahesh@/opt$:
```

- In a root terminal, type

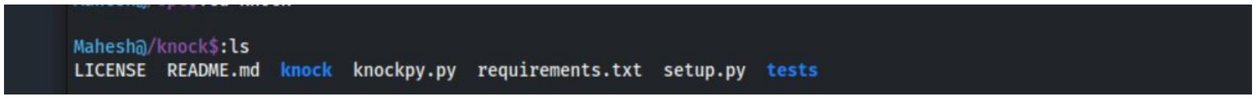
  `git clone https://github.com/guelfoweb/knock`

knock



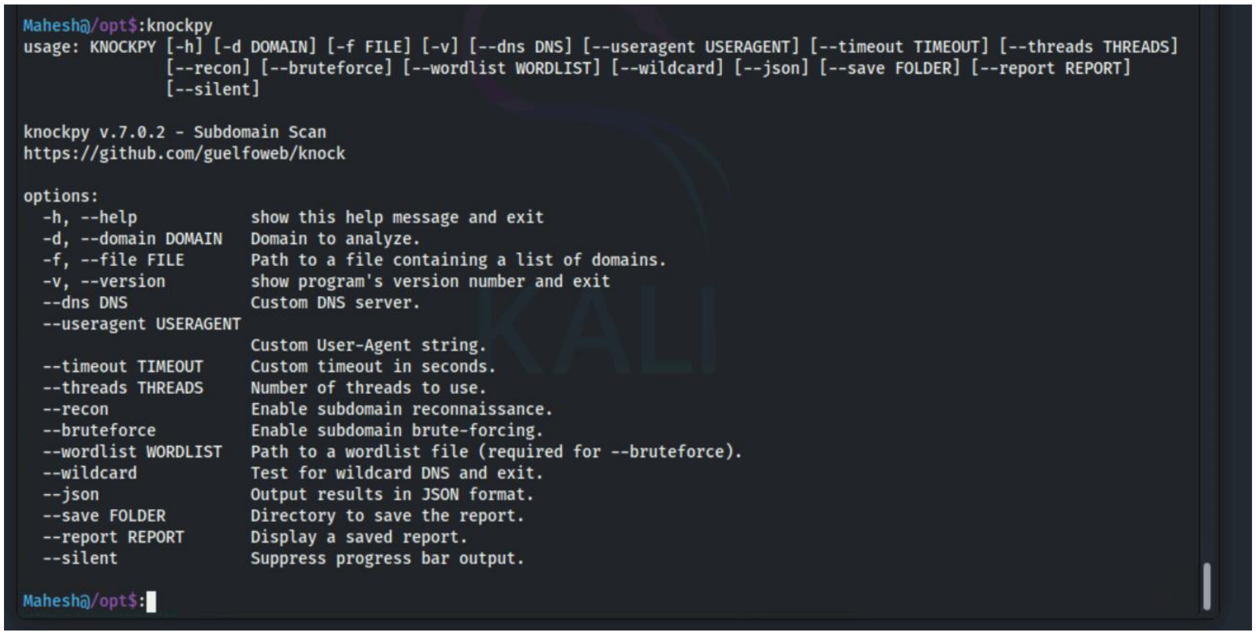
```
Mahesh@/opt$:git clone https://github.com/guelfoweb/knock
Cloning into 'knock'...
remote: Enumerating objects: 1633, done.
remote: Counting objects: 100% (59/59), done.
remote: Compressing objects: 100% (44/44), done.
remote: Total 1633 (delta 26), reused 16 (delta 15), pack-reused 1574 (from 3)
Receiving objects: 100% (1633/1633), 553.41 KiB | 13.50 MiB/s, done.
Resolving deltas: 100% (745/745), done.
```

- Navigate to the **knock** directory type **`python3 setup.py install`** and press enter.



```
Mahesh@/knock$:ls
LICENSE  README.md  knock  knockpy.py  requirements.txt  setup.py  tests
```

- To test the installation, type **`knockpy`** and press enter.



```
Mahesh@/opt$:knockpy
usage: KNOCKPY [-h] [-d DOMAIN] [-f FILE] [-v] [--dns DNS] [--useragent USERAGENT] [--timeout TIMEOUT] [--threads THREADS]
               [--recon] [--bruteforce] [--wordlist WORDLIST] [--wildcard] [--json] [--save FOLDER] [--report REPORT]
               [--silent]

knockpy v.7.0.2 - Subdomain Scan
https://github.com/guelfoweb/knock

options:
  -h, --help            show this help message and exit
  -d, --domain DOMAIN   Domain to analyze.
  -f, --file FILE       Path to a file containing a list of domains.
  -v, --version         show program's version number and exit
  --dns DNS             Custom DNS server.
  --useragent USERAGENT
                        Custom User-Agent string.
  --timeout TIMEOUT     Custom timeout in seconds.
  --threads THREADS     Number of threads to use.
  --recon               Enable subdomain reconnaissance.
  --bruteforce          Enable subdomain brute-forcing.
  --wordlist WORDLIST   Path to a wordlist file (required for --bruteforce).
  --wildcard            Test for wildcard DNS and exit.
  --json                Output results in JSON format.
  --save FOLDER         Directory to save the report.
  --report REPORT       Display a saved report.
  --silent              Suppress progress bar output.

Mahesh@/opt$:
```

In this lesson, you have learned how to install several web application testing tools. In the next lesson, we will see how each of these tools can help streamline web application reconnaissance.