

# Top 5 Tools & Tricks for Ethical Hacking & Bug Bounties 2021

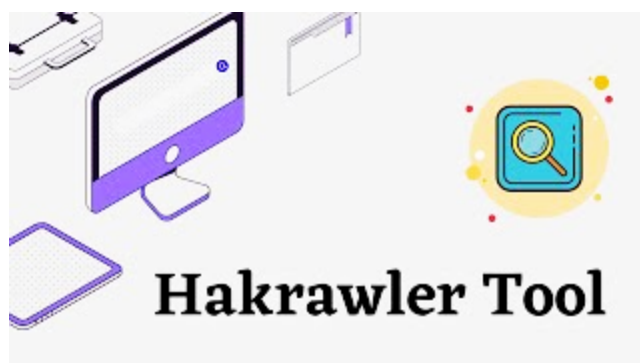
**Unveiling the Ultimate Toolkit: Mastering Ethical Hacking & Bug Bounties with Top 5 Tools & Tricks 2021**



## **Introduction:**

In the dynamic landscape of cybersecurity, staying one step ahead of potential threats is paramount. Ethical hackers, penetration testers, and bug bounty hunters are the guardians of this digital realm, armed with knowledge, skills, and cutting-edge tools. Welcome to the transformative Udemy course "Top 5 Tools & Tricks for Ethical Hacking & Bug Bounties 2021." In this article, we'll provide you with a glimpse into the exciting world of this course, which unveils the quintessential tools and techniques that can empower you to become a skilled cybersecurity professional.

## **Introduction to Hakrawler:-**



Hakrawler is a versatile open-source tool for web application security testing, designed to uncover concealed files and directories on web servers. This command-line utility is compatible with Linux, macOS, and Windows operating systems, making it accessible to a wide range of users. In this blog post, we'll delve into the key features of Hakrawler and provide a comprehensive guide on how to effectively utilize this tool for web application security assessments.

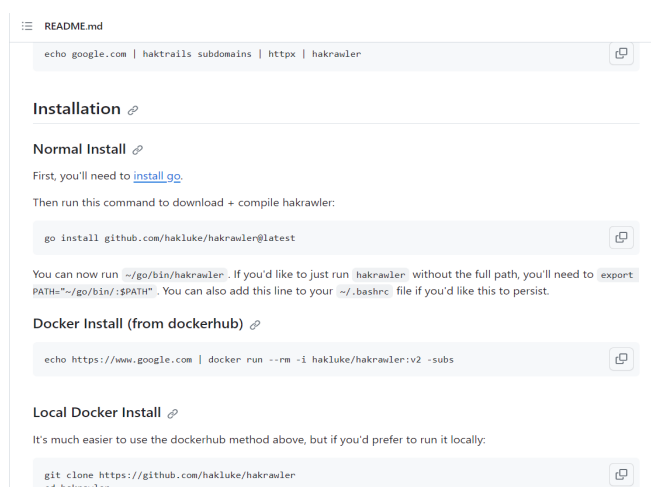
## Features of Hakrawler

Hakrawler has the following features:

- **Discover hidden files and directories:** Hakrawler can be used to discover hidden files and directories on a web server. This can be useful for web application security testing.
- **Fast and efficient:** Hakrawler is fast and efficient. It can crawl a website quickly and discover hidden files and directories.
- **Command-line tool:** Hakrawler is a command-line tool that can be used on Linux, macOS, and Windows.
- **Customizable:** Hakrawler is customizable. It can be configured to ignore certain files and directories.

## Installation Process

→ **Go to this link and scroll down to the installation part and copy the command**



```
echo google.com | haktrails subdomains | httpx | hakrawler
```

### Installation

#### Normal Install

First, you'll need to [install go](#).

Then run this command to download + compile hakrawler:

```
go install github.com/hakluke/hakrawler@latest
```

You can now run `~/go/bin/hakrawler`. If you'd like to just run `hakrawler` without the full path, you'll need to `export PATH="$~/go/bin:$PATH"`. You can also add this line to your `~/bashrc` file if you'd like this to persist.

#### Docker Install (from dockerhub)

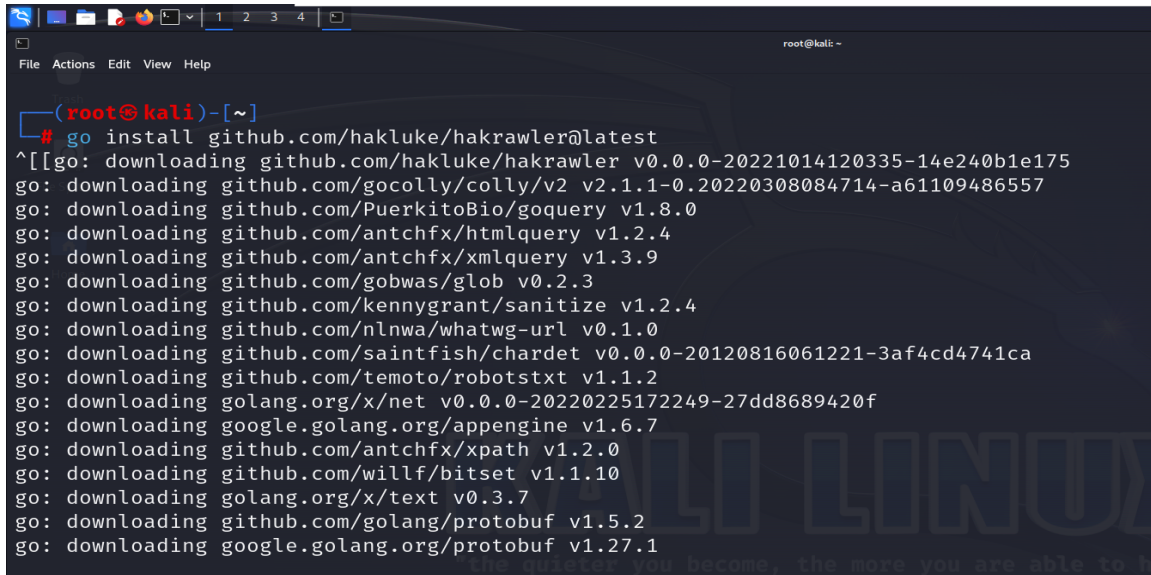
```
echo https://www.google.com | docker run --rm -i hakluke/hakrawler:v2 -subs
```

#### Local Docker Install

It's much easier to use the dockerhub method above, but if you'd prefer to run it locally:

```
git clone https://github.com/hakluke/hakrawler
cd hakrawler
```

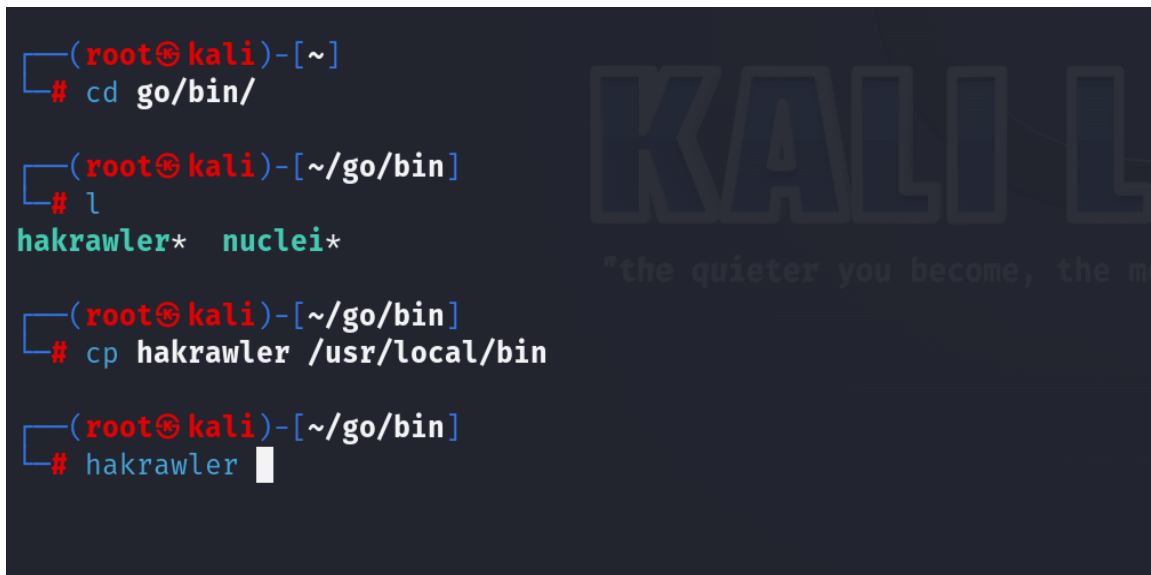
→ Open the terminal and paste the code



```
(root@kali)-[~]
└─# go install github.com/hakluke/hakrawler@latest
^[[go: downloading github.com/hakluke/hakrawler v0.0.0-20221014120335-14e240b1e175
go: downloading github.com/gocolly/colly/v2 v2.1.1-0.20220308084714-a61109486557
go: downloading github.com/PuerkitoBio/goquery v1.8.0
go: downloading github.com/antchfx/htmlquery v1.2.4
go: downloading github.com/antchfx/xmlquery v1.3.9
go: downloading github.com/gobwas/glob v0.2.3
go: downloading github.com/kennygrant/sanitize v1.2.4
go: downloading github.com/nlnwa/whatwg-url v0.1.0
go: downloading github.com/saintfish/chardet v0.0.0-20120816061221-3af4cd4741ca
go: downloading github.com/temoto/robotstxt v1.1.2
go: downloading golang.org/x/net v0.0.0-20220225172249-27dd8689420f
go: downloading google.golang.org/appengine v1.6.7
go: downloading github.com/antchfx/xpath v1.2.0
go: downloading github.com/willf/bitset v1.1.10
go: downloading golang.org/x/text v0.3.7
go: downloading github.com/golang/protobuf v1.5.2
go: downloading google.golang.org/protobuf v1.27.1
```

Figure:- The above figure shows the installation process

→ Then to `cd /go/bin/` after that copy the tool to `/usr/local/bin/`



```
(root@kali)-[~]
└─# cd go/bin/

(root@kali)-[~/go/bin]
└─# ls
hakrawler*  nuclei*

(root@kali)-[~/go/bin]
└─# cp hakrawler /usr/local/bin

(root@kali)-[~/go/bin]
└─# hakrawler
```

Figure:- The above figure shows the copying the tool to `/usr/local/bin`

## → Help menu of the tool

```
(root@kali)-[~]
└─# hakrawler -h
flag needs an argument: -h
Usage of hakrawler:
-d int
  Depth to crawl. (default 2)
-dr
  Disable following HTTP redirects.
-h string
  Custom headers separated by two semi-colons. E.g. -h "Cookie: foo=bar;;Referer: http://example.com/"
-i
  Only crawl inside path
-insecure
  Disable TLS verification.
-json
  Output as JSON.
-proxy string
  Proxy URL. E.g. -proxy http://127.0.0.1:8080
-s
  Show the source of URL based on where it was found. E.g. href, form, script, etc.
-size int
  Page size limit, in KB. (default -1)
-subs
  Include subdomains for crawling.
-t int
  Number of threads to utilise. (default 8)
-timeout int
  Maximum time to crawl each URL from stdin, in seconds. (default -1)
-u
  Show only unique urls.
-w
  Show at which link the URL is found.

(root@kali)-[~]
└─#
```

Figure:- The above figure shows the flags of the tool

## → Output

```
(root@kali)-[~]
└─# echo "https://testphp.com" | hakrawler
https://testphp.com/px.js?ch=1&abp=1
https://testphp.com/px.js?ch=2&abp=1

(root@kali)-[~]
└─# echo "https://example.com" | hakrawler
https://www.iana.org/domains/example

(root@kali)-[~]
└─#
```

Figure:- The above figure shows the output of Hakrawler running on testphp.com and example.com

## → Reference:-

1. <https://github.com/hakluke/hakrawler>