2 D Moon

Subdomain/DNS Enumeration

Gaw

Group 3

Herrera

King

Muros

Panahon

brief background of

DNS

DNS servers form the heart and soul of the internet. Without it, we'd be stuck with inputting long lists of numbers (IP addresses) to access the content we want.

Because of this, it is one of the most frequently attacked protocols where malicious actors attempt to create a full inventory of all internet-connected devices and domain names from the target company's domain.

brief background of

DNS

The process of discovering sub-domains for one or more domains is known as subdomain enumeration. It aids in broadening the attack surface, and discovering hidden applications and forgotten subdomains.

Introduction of

2 D Moon

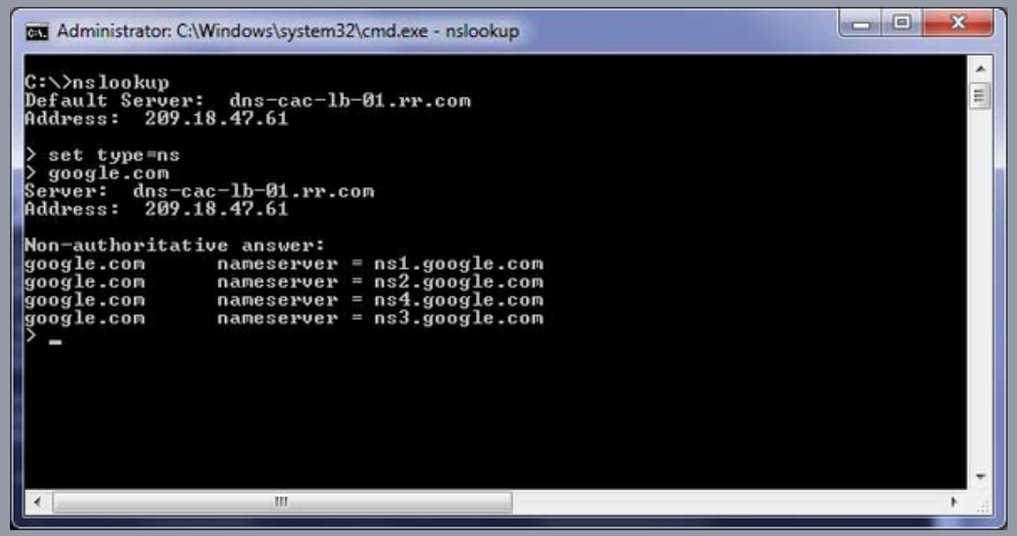
2 D Moon is a CLI-based tool, written in Python, that receives input commands from a user in the form of lines of text and output results in the same format.

It is meant to be a beginner's tool that mainly caters to those who just started taking cybersecurity courses, or simply curious cats wanting to explore a different world from their own.

Aside from being a tool that does the job done, 2 D Moon is also a learning-centered tool wherein a guide on how to use the publicly available tools to reproduce the same results is given.

DNS LOOKUP

NSLOOKUP



- stands for “Name Server Lookup"
- one of the oldest tools
- available on all major OS
- when making DNS queries, does not use the local resolver provided by the operating system, instead uses its own internal resolver
- this leads to inconsistency and sometimes confusing behavior in the results
- not recommended because of far better choices

Usage: nslookup [host server]
Availability: Cross-platform software

DNS LOOKUP

HOST

```
┌──(kali㉿kali)-[~]
└─$ host google.com
google.com has address 172.217.24.110
google.com has IPv6 address 2404:6800:4005:800::200e
google.com mail is handled by 10 smtp.google.com.

┌──(kali㉿kali)-[~]
└─$ host 172.217.24.110
110.24.217.172.in-addr.arpa domain name pointer sin10s07-in-f14.1e100.net.
110.24.217.172.in-addr.arpa domain name pointer hkg12s33-in-f14.1e100.net.
110.24.217.172.in-addr.arpa domain name pointer sin10s07-in-f110.1e100.net.
```

- a simple and quick utility; doesn't have a lot of functions
- usually used for converting domain names to IP addresses and the other way around
- provide functions for reverse lookups as well as DNS queries via IPv6 networking

Usage: host [hostname]
Availability: Unix, Unix-like

DNS LOOKUP

DIG

```
┌──(kali㉿kali)-[~]
└─$ dig google.com

; <<>> DiG 9.18.1-1-Debian <<>> google.com
;; global options: +cmd
;; Got answer:
;; →→HEADER←← opcode: QUERY, status: NOERROR, id: 2714
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; MBZ: 0×0005, udp: 4096
;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             5       IN      A       172.217.24.110

;; Query time: 7 msec
;; SERVER: 192.168.30.2#53(192.168.30.2) (UDP)
;; WHEN: Thu Dec 08 11:53:31 EST 2022
;; MSG SIZE  rcvd: 55
```

- stands for "Domain Information Groper"
- basically an nslookup that's more flexible, functional, and reliable
- dig defaults to querying for DNS results according to your computer's network settings
- capable of querying specific name servers and even of tracing through the resolution of a particular query
- recommended over other local-based tools and replaces older tools such as nslookup and host

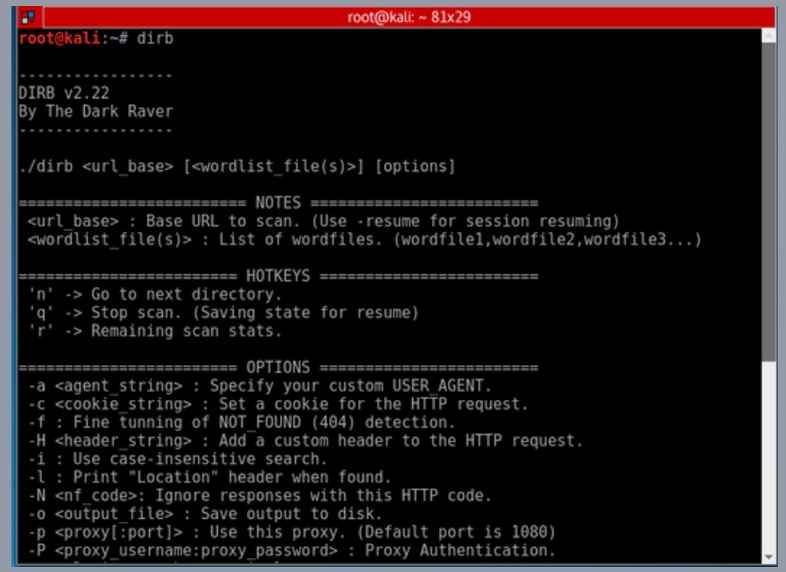
Usage: dig [host]
Availability: Linux, NetBSD, FreeBSD, OpenBSD, macOS, Windows, Solaris, illumos, OpenVMS

SUBDOMAIN ENUMERATION

DIRB

- written in C language
- comes preinstalled with a set of preconfigured attack wordlists for easy usage, but allows the addition of custom wordlists
- usually utilized in Web Application testing or Auditing
- covers some holes not covered by classic web vulnerability scanners
- is a content scanner not a vulnerability scanner; it doesn't search vulnerabilities nor does it look for web contents that can be vulnerable

Usage: dirb [web server]

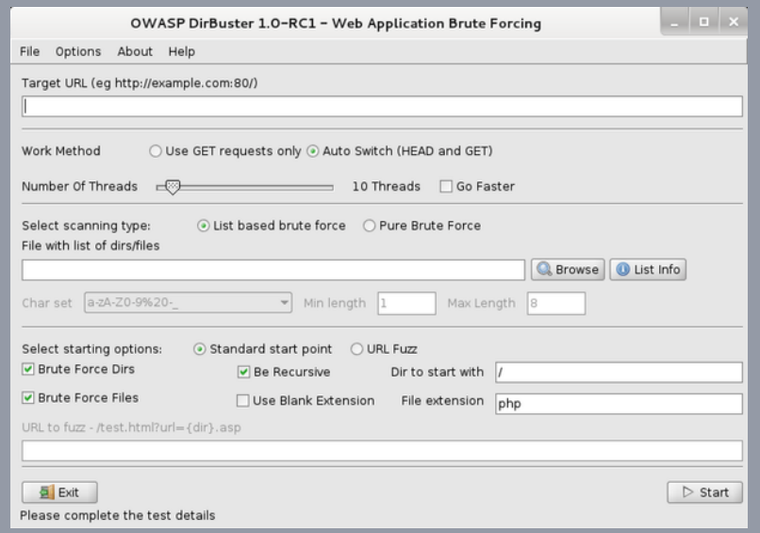


SUBDOMAIN ENUMERATION

- written in Java
- offers a GUI interface
- the effectiveness lies on the wordlist, the more effective the wordlist, the more effective the tool
- unlike Gobuster, DirBuster can detect very deep nesting of subpages with a single command

Usage: enter the requested inputs then click start

DIRBUSTER

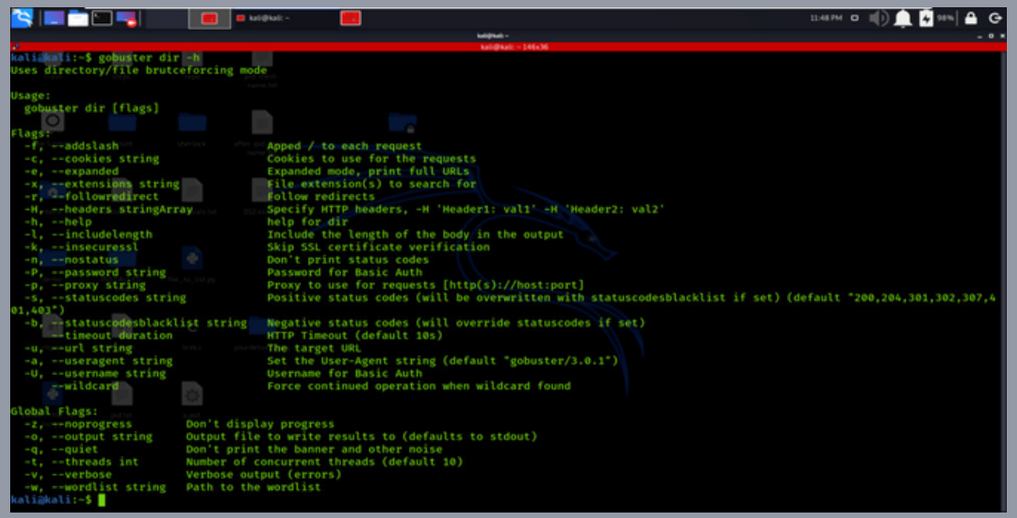


SUBDOMAIN ENUMERATION

GOBUSTER

- written in Go language
- can search virtual host names on target web servers
- The main advantage of Gobuster is the lighting speed faster performance
- in popular directories, where DirBuster and DIRB would be slow and responsive to errors, Gobuster will thrive
- lack of recursive directory exploration/searching; for directories more than one level deep, another scan will be needed

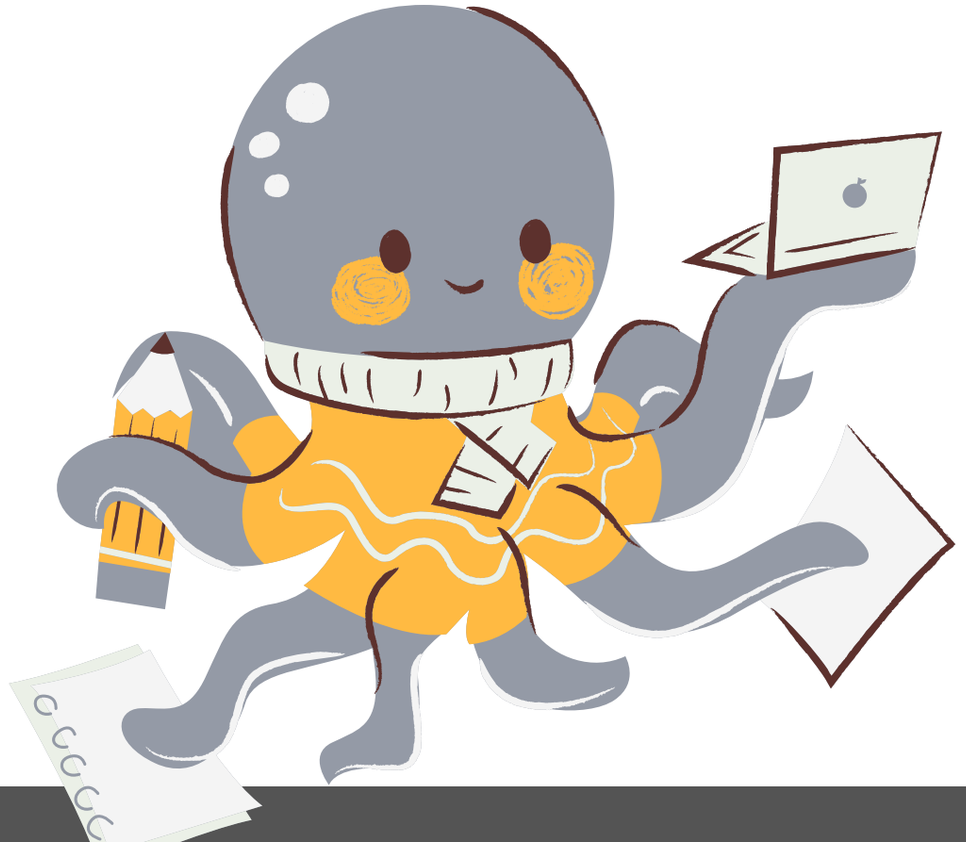
Usage: gobuster [command]



2 D Moon is...

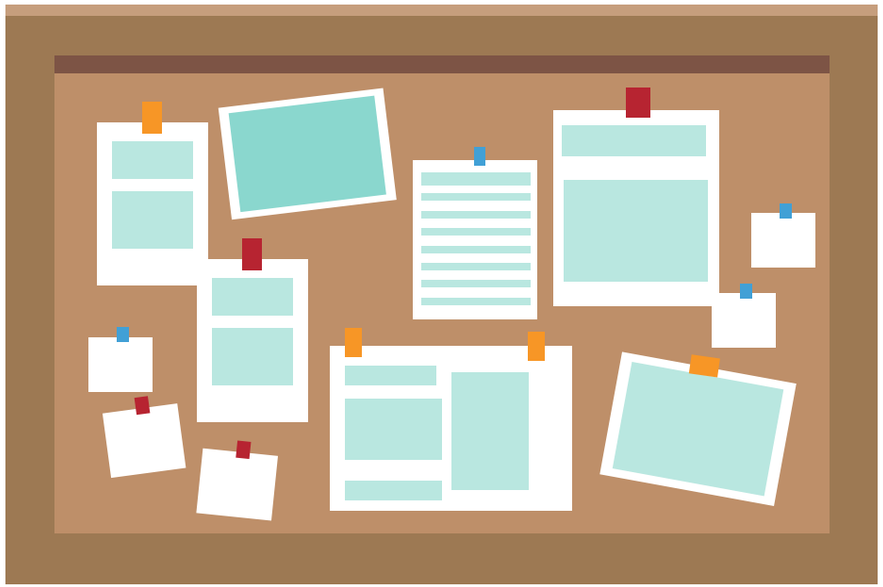
2-in-1

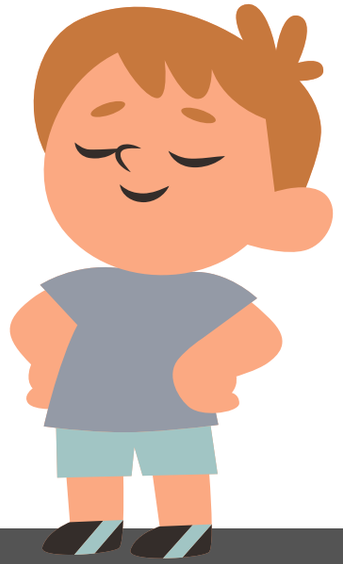
No need for switching between tools ! 2 D Moon performs both **DNS footprinting** and **Subdomain Enumeration**.



As easy as pie

No memorization for commands and options required!
We'll do the nitty-gritty for you,
Just Type Stuff, and See Stuff.





A Learner's Tool

Aside from being a hacking tool, 2 D Moon provides a cheat sheet on the **basics of DNSing**. On it are the how-tos of the de-facto standard tools, and a list of online resources relevant for both **beginners and intermediate** alike.

Highly Customizable

2 D Moon being an open-source software is the closest as you can get to a Filipino home. This allows the customization of the product to fit your personal needs, or the organization's needs.

2 D Moon Tool Demonstration and Code Review

DNS Enumeration

```
def single_req(type):
    print("\n" + type + " record")
    print("-"*30)
    try:
        answer = dns.resolver.resolve(link,type,raise_on_no_answer=False)

        if answer.rrset is not None:
            print(answer.rrset)
    except:
        pass
```

DNS Enumeration

```
Instruction: 1
Enter Domain: youtube.com
Record Type: all
A record
-----------------------------
youtube.com. 72 IN A 172.217.24.78


AAAA record
-----------------------------
youtube.com. 103 IN AAAA 2404:6800:4005:81c::200e


MX record
-----------------------------
youtube.com. 238 IN MX 0 smtp.google.com.


NS record
-----------------------------
youtube.com. 138325 IN NS ns2.google.com.
youtube.com. 138325 IN NS ns1.google.com.
youtube.com. 138325 IN NS ns3.google.com.
youtube.com. 138325 IN NS ns4.google.com.


CNAME record
-----------------------------
SOA record
-----------------------------
youtube.com. 60 IN SOA ns1.google.com. dns-admin.google.com. 493848506 900 900 1800 60


TXT record
-----------------------------
youtube.com. 1821 IN TXT "google-site-verification=QtQWEwHWM8tHiJ4s-jJWzEQrD_fF3luPnpzNDH-Nw-w"
youtube.com. 1821 IN TXT "facebook-domain-verification=64jdes7le4h7e7lfpi22rijygx58j1"
youtube.com. 1821 IN TXT "v=spf1 include:google.com mx -all"
```

Subdomain Enumeration

```python
def subdomain_scan():
    inp_list = input("Use a different wordlist? ").lower()

    if inp_list == "y":
        list = input("Enter wordlist file: ")
        sub_list = open(list).read()
        subdomain_array = sub_list.splitlines()
    else:
        subdomain_array = ['www', 'mail', 'ftp', 'localhost', 'webmail',

    subdomain_store = []
    for subdoms in subdomain_array:
        try:
            ip_value = dns.resolver.resolve(f'{subdoms}.{link}', 'A')
            if ip_value:
                subdomain_store.append(f'{subdoms}.{link}')
                if f"{subdoms}.{link}" in subdomain_store:
                    print(f'{subdoms}.{link} valid')
                else:
                    pass
        except:
            pass
```

Subdomain Enumeration

w/o wordlist

```
Instruction: 2
Enter Domain: google.com
Use a different wordlist? n
www.google.com valid
mail.google.com valid
smtp.google.com valid
ns1.google.com valid
ns2.google.com valid
m.google.com valid
ns.google.com valid
blog.google.com valid
admin.google.com valid
news.google.com valid
vpn.google.com valid
ns3.google.com valid
support.google.com valid
mobile.google.com valid
docs.google.com valid
calendar.google.com valid
web.google.com valid
email.google.com valid
images.google.com valid
video.google.com valid
api.google.com valid
ns4.google.com valid
dns.google.com valid
search.google.com valid
chat.google.com valid
wap.google.com valid
sites.google.com valid
ads.google.com valid
```

Subdomain Enumeration

w/ wordlist

```
Instruction: 2
Enter Domain: google.com
Use a different wordlist? wordlist.txt
www.google.com valid
mail.google.com valid
smtp.google.com valid
ns1.google.com valid
ns2.google.com valid
m.google.com valid
ns.google.com valid
blog.google.com valid
admin.google.com valid
news.google.com valid
vpn.google.com valid
ns3.google.com valid
support.google.com valid
mobile.google.com valid
docs.google.com valid
calendar.google.com valid
web.google.com valid
email.google.com valid
images.google.com valid
video.google.com valid
api.google.com valid
ns4.google.com valid
dns.google.com valid
search.google.com valid
chat.google.com valid
wap.google.com valid
sites.google.com valid
ads.google.com valid
```

Reverse DNS Lookup

```
def get_domname(IP):
    addrs = dns.reversename.from_address(IP)
    try:
        print("Domain name of " + str(IP) + " is " + str(dns.resolver.resolve(addrs,"PTR")[0]))
    except:
        print("Does not exist")
```

Reverse DNS Lookup

```
Instruction: 3
Enter IP: 157.240.199.35
Domain name of 157.240.199.35 is edge-star-mini-shv-01-hkg4.facebook.com.
-----------------------------
```

Reverse DNS Lookup

nslookup

```
C:\Users\julia>nslookup facebook.com
Server:  UnKnown
Address:  172.20.10.1

Non-authoritative answer:
Name:    facebook.com
Addresses:  2a03:2880:f15e:83:face:b00c:0:25de
          157.240.199.35
```

DNS Zone Transfer

```python
def dns_zone_xfer(address):
    ns_servers = []
    try:
        ns_answer = dns.resolver.resolve(address, 'NS')
        if ns_answer.rrset is not None:
            for server in ns_answer:
                print("[*] Found NS: {}".format(server))
                ip_answer = dns.resolver.resolve(server.target, 'A')
                for ip in ip_answer:
                    print("[*] IP for {} is {}".format(server, ip))
                    try:
                        zone = dns.zone.from_xfr(dns.query.xfr(str(ip), address))
                        for host in zone:
                            print("[*] Found Host: {}".format(host))
                    except Exception as e:
                        print("[*] NS {} refused zone transfer!".format(server))
                        continue
    except dns.resolver.NXDOMAIN:
        print("\nDomain does not exist \n")
    except:
        pass
```

DNS Zone Transfer

```
1 - DNS Enumeration
2 - Subdomain Enumeration
3 - Reverse DNS Lookup
4 - DNS Zone Transfer
5 - DNS Cheat Sheet
6 - EXIT
Instruction: 4
Enter Domain: gooogle.com
[*] Found NS: ns1.google.com.
[*] IP for ns1.google.com. is 216.239.32.10
[*] NS ns1.google.com. refused zone transfer!
[*] Found NS: ns3.google.com.
[*] IP for ns3.google.com. is 216.239.36.10
[*] NS ns3.google.com. refused zone transfer!
[*] Found NS: ns2.google.com.
[*] IP for ns2.google.com. is 216.239.34.10
[*] NS ns2.google.com. refused zone transfer!
[*] Found NS: ns4.google.com.
[*] IP for ns4.google.com. is 216.239.38.10
[*] NS ns4.google.com. refused zone transfer!
```

```
1 - DNS Enumeration
2 - Subdomain Enumeration
3 - Reverse DNS Lookup
4 - DNS Zone Transfer
5 - DNS Cheat Sheet
6 - EXIT
Instruction: 4
Enter Domain: zonetransfer.me
[*] Found NS: nsztm2.digi.ninja.
[*] IP for nsztm2.digi.ninja. is 34.225.33.2
[*] Found Host: @
[*] Found Host: _acme-challenge
[*] Found Host: _sip._tcp
[*] Found Host: 14.105.196.5.IN-ADDR.ARPA
[*] Found Host: asfdbauthdns
[*] Found Host: asfdbbox
[*] Found Host: asfdbvolume
[*] Found Host: canberra-office
[*] Found Host: cmdexec
[*] Found Host: contact
[*] Found Host: dc-office
[*] Found Host: deadbeef
[*] Found Host: dr
[*] Found Host: DZC
[*] Found Host: email
[*] Found Host: Hello
[*] Found Host: home
[*] Found Host: Info
[*] Found Host: internal
[*] Found Host: intns1
[*] Found Host: intns2
[*] Found Host: office
[*] Found Host: ipv6actnow.org
[*] Found Host: owa
[*] Found Host: robinwood
[*] Found Host: rp
[*] Found Host: sip
[*] Found Host: sqli
```