main | Branches | Tags | Go to file | Code

.github
asn
cmd
integration_tests
static
.gitignore
.goreleaser.yml
Dockerfile
LICENSE.MD
Makefile
README.md
cidr.go
error.go
go.mod
go.sum
ip.go
ip_test.go
shuffle.go
type.go
urlencode.go
utils.go

About

Utility program to perform multiple operations for a given subnet/CIDR ranges.

projectdiscovery.io

#cidr #subnetting #subnetwork #cidr-ranges

Readme
MIT license
Code of conduct
Security policy
Activity
Custom properties
952 stars
27 watching
93 forks
Report repository

Releases 31

v1.1.34 Latest
on Apr 2

+ 30 releases

Packages

No packages published

Used by 553

+ 545

Contributors 18

+ 4 contributors

Languages

Go 98.9% Other 1.1%

README | Code of conduct | More

mapCIDR

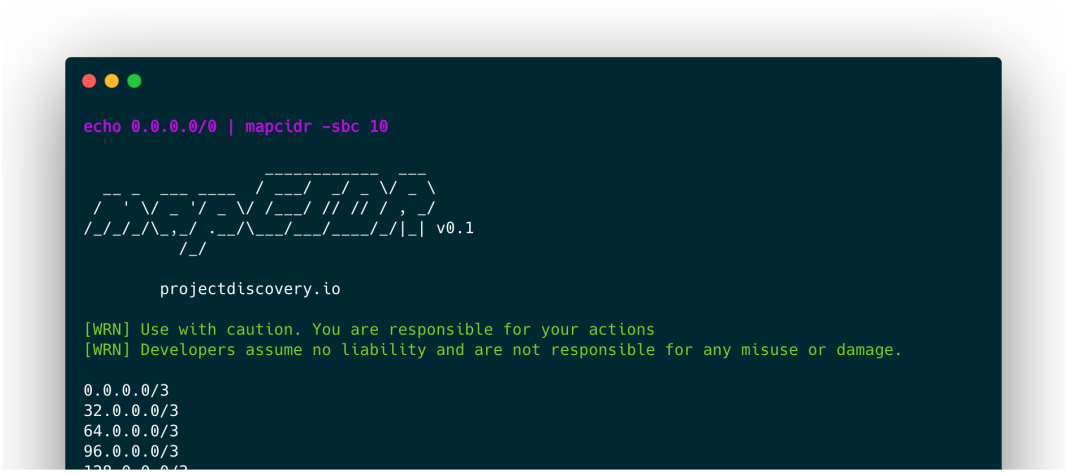
A utility program to perform multiple operations for a given subnet/cidr ranges.

contributions welcome | release v1.1.34 | Follow @pdiscoveryio | chat 835 online

Features • Install • Usage • Library • Join Discord

mapCIDR is developed to ease load distribution for mass scanning operations, it can be used both as a library and as independent CLI tool.

Features



- **CIDR expansion** support (**default**)
- **CIDR slicing** support (`sbh` , `sbc`)
- **CIDR/IP aggregation** support (`a` , `aa`)
- **CIDR/IP matcher** support (`match-ip`)
- **CIDR/IP filter** support (`filter-ip`)
- **CIDR/IP sorting** support (`s` , `sr`)
- **CIDR host count** support (`count`)
- Multiple **IP Format** support (`ip-format`)
- IP/PORT shuffling support (`si` , `sp`)
- **IPv4/IPv6 Conversation** support (`t4` , `t6`)
- CIDR STDIN (pipe) input support

Installation

```
go install -v github.com/projectdiscovery/mapcidr/cmd/mapcidr@latest
```

Usage

```
mapcidr -h
```

This will display help for the tool. Here are all the switches it supports.

```
INPUT:
   -cl, -cidr string[]  CIDR/IP/File containing list of CIDR/IP to process

PROCESS:
   -sbc int                 Slice CIDRs by given CIDR count
   -sbh int                 Slice CIDRs by given HOST count
   -a, -aggregate           Aggregate IPs/CIDRs into minimum subnet
   -aa, -aggregate-approx   Aggregate sparse IPs/CIDRs into minimum approximated subnet
   -c, -count               Count number of IPs in given CIDR
```

```
   -t4, -to-ipv4             Convert IPs to IPv4 format
   -t6, -to-ipv6             Convert IPs to IPv6 format
   -ip-format, -if string[]  IP formats (0,1,2,3,4,5,6,7,8,9,10,11)
   -zpn, -zero-pad-n int     number of padded zero to use (default 3)
   -zpp, -zero-pad-permute   enable permutations from 0 to zero-pad-n for each octets

FILTER:
   -f4, -filter-ipv4         Filter IPv4 IPs from input
   -f6, -filter-ipv6         Filter IPv6 IPs from input
   -skip-base                Skip base IPs (ending in .0) in output
   -skip-broadcast           Skip broadcast IPs (ending in .255) in output
   -mi, -match-ip string[]   IP/CIDR/FILE containing list of IP/CIDR to match (comma-separated, file input)
   -fi, -filter-ip string[]  IP/CIDR/FILE containing list of IP/CIDR to filter (comma-separated, file input)

MISCELLANEOUS:
   -s, -sort                  Sort input IPs/CIDRs in ascending order
   -sr, -sort-reverse         Sort input IPs/CIDRs in descending order
   -si, -shuffle-ip           Shuffle Input IPs in random order
   -sp, -shuffle-port string  Shuffle Input IP:Port in random order

UPDATE:
   -up, -update                 update mapcidr to latest version
   -duc, -disable-update-check  disable automatic mapcidr update check

OUTPUT:
   -verbose            Verbose mode
   -o, -output string  File to write output to
   -silent             Silent mode
   -version            Show version of the project
```

Running mapCIDR

In order to get list of IPs for a give CIDR, use the following command.

CIDR expansion

```
mapcidr -cidr 173.0.84.0/24
```

```
                  ____________  ___
  __ _  ___ ____  / ___/  _/ _ \/ _ \
 /  ' \/ _ '/ _ \/ /___/ // // / , _/
/_/_/_/\_,_/ .__/\___/___/____/_/|_| v0.5
          /_/

               projectdiscovery.io

[WRN] Use with caution. You are responsible for your actions
[WRN] Developers assume no liability and are not responsible for any misuse or damage.

173.0.84.0
173.0.84.1
173.0.84.2
173.0.84.3
173.0.84.4
173.0.84.5
173.0.84.13
173.0.84.14
173.0.84.15
173.0.84.16
```

It is also possible to get list of IP's for a given IP range, use the following command

```
$ echo "192.168.0.0-192.168.0.5" | mapcidr
```

```
192.168.0.0
192.168.0.1
192.168.0.2
```

```
192.168.0.3
192.168.0.4
192.168.0.5
```

CIDR Slicing by CIDR Count

In order to slice given CIDR or list of CIDR by CIDR count or slice into multiple and equal smaller subnets, use the following command.

```
mapcidr -cidr 173.0.84.0/24 -sbc 10 -silent
```

```
173.0.84.0/27
173.0.84.32/27
173.0.84.64/27
173.0.84.96/27
173.0.84.128/27
173.0.84.160/27
173.0.84.208/28
173.0.84.192/28
173.0.84.240/28
173.0.84.224/28
```

CIDR slicing by HOST Count

In order to slice given CIDR for equal number of host count in each CIDR, use the following command.

```
mapcidr -cidr 173.0.84.0/16 -sbh 20000 -silent
```

```
173.0.0.0/18
173.0.64.0/18
173.0.128.0/18
173.0.192.0/18
```

Note: it's possible to obtain a perfect split only when the desired amount of slices or hosts per subnet is a powers of two. Otherwise, the tool will attempt to automatically find the best split strategy to obtain the desired outcome.

CIDR/IP Aggregation

In order to merge multiple CIDR ranges into smaller subnet block, use the following command.

```
$ mapcidr -cl cidrs.txt -aggregate
```

In order to list CIDR blocks for given list of IPs, use the following command.

```
$ mapcidr -il ips.txt -aggregate
```

It's also possible to perform approximated aggregations for sparse ips groups (only version 4). The final interval will contain contiguous ips not belonging to the input:

```
$ cat ips.txt

1.1.1.1
1.1.1.16
1.1.1.31
```

```
$ cat ips.txt | mapcidr -aggregate-approx

1.1.1.0/27
```

In order to list CIDR blocks for given IP Range (**IPv4** | **IPv6**), use the following command.

```
$ mapcidr  -cl 192.168.0.1-192.168.0.255 -aggregate
OR
$ echo 192.168.0.1-192.168.0.255 | mapcidr -aggregate
```

```
192.168.0.1/32
192.168.0.2/31
192.168.0.4/30
192.168.0.8/29
192.168.0.16/28
192.168.0.32/27
192.168.0.64/26
192.168.0.128/25
```

Match / Filter IP's from CIDR

In order to match IPs from the given list of CIDR ranges, use the following command.

```
$ mapcidr -cidr 192.168.1.0/24 -mi 192.168.1.253,192.168.1.252
$ mapcidr -cidr 192.168.1.0/24 -mi ip_list_to_match.txt
```

In order to match IPs from the given list of CIDR ranges, use the following command.

```
$ mapcidr -cidr 192.168.1.224/28 -fi 192.168.1.233,192.168.1.234
$ mapcidr -cidr 192.168.1.224/28 -fi ip_list_to_filter.txt
```

IP Formats

In order to represent given IP into multiple formats, `-if 0` flag can be used to display all the supported format values, and specific type of format can be displayed using specific index number as listed here, currently 10 unique formats are supported.

```
$ echo 127.0.1.0 | mapcidr -if 0 -silent

127.0.1.0
127.1
0177.0.01.0
0x7f.0x0.0x1.0x0
0x7f000100
0xabfa659dfa7f000100
281472812450048
11111111111111111011111110000000000000000100000000
0x7f.0.01.0x0
::ffff:7f00:0100
%31%32%37%2E%30%2E%31%2E%30
127.000.001.000
```

IP Conversion

IPv4 | **IPv6** addresses can be converted from either the v6 to v4 notation or IPv4-mapped notation into IPv4 addresses using `-t4` and `-t6` to IPv4 and IPv6 respectively.

```
$ cat ips.txt

1.1.1.1
2.2.2.2
```

```
$ mapcidr -cl ips.txt -t6

00:00:00:00:00:ffff:0101:0101
00:00:00:00:00:ffff:0202:0202
```

> **Note:**
>
> Not all IPv6 address can be converted to IPv4. You can only convert valid IPv4 represented IPv6 addresses.

CIDR Host Counting

In order to count number of hosts for a given CIDR or list of CIDR, use the following command.

```
$ echo 173.0.84.0/16 | mapcidr -count -silent

65536
```

ASN Input

In order to get the IP address of ASN number, use the following command

```
echo AS15133 | mapcidr -silent

5.104.64.0
5.104.64.1
5.104.64.2
5.104.64.3
5.104.64.4
```

Use mapCIDR as a library

It's possible to use the library directly in your go programs. The following code snippets outline how to divide a cidr into subnets, and how to divide the same into subnets containing a certain number of hosts

```
package main

import (
        "fmt"

        "github.com/projectdiscovery/mapcidr"
)

func main() {
        // Divide the CIDR into two subnets
        subnets1 := mapcidr.SplitN("192.168.1.0/24", 2)
        for _, subnet := range subnets1 {
                fmt.Println(subnet)
        }
        // Divide the CIDR into two subnets containing 128 hosts each
        subnets2 := mapcidr.SplitByNumber("192.168.1.0/24", 128)
        for _, subnet := range subnets2 {
                fmt.Println(subnet)
        }

        // List all ips in the CIDR
        ips, _ := mapcidr.IPAddresses("192.168.1.0/24")
        for _, ip := range ips {
                fmt.Println(ip)
        }
```