Università di Messina — CdL in Informatica — Reti e Sistemi Distribuiti Mod.B

Guida a `ipcalc-ng`

Calcolo e analisi di subnet IPv4 e IPv6 da riga di comando

Last Update: March 26, 2026

Introduzione

`ipcalc-ng` è uno strumento da riga di comando per il calcolo e l'analisi di indirizzi IP e subnet, sia IPv4 che IPv6. È il successore moderno e attivamente mantenuto del classico `ipcalc` originario delle distribuzioni Red Hat.

[!] Due strumenti con lo stesso nome

Esistono due progetti distinti entrambi chiamati `ipcalc`:

- **ipcalc classico** (Red Hat/CentOS/Fedora): non supporta IPv6, non più mantenuto attivamente.
- **ipcalc-ng** (Ubuntu/Debian): supporto IPv6, output JSON, GeoIP, split, deaggregate. Questo è il documento di riferimento.

Su Ubuntu/Debian il pacchetto si installa con `apt install ipcalc-ng`. Il binario potrebbe chiamarsi `ipcalc` a seconda della distribuzione: verifica con `which ipcalc ipcalc-ng`.

Analisi base di una subnet

La forma più semplice di utilizzo: fornire un indirizzo IP con il suo prefisso CIDR.

```
ipcalc-ng 192.168.1.47/24
```

```
Address:          192.168.1.47
Network:          192.168.1.0/24
Netmask:          255.255.255.0 = 24
Broadcast:        192.168.1.255

Address space:    Private Use
HostMin:          192.168.1.1
HostMax:          192.168.1.254
Hosts/Net:        254
```

L'output mostra immediatamente i campi fondamentali della subnet. Il campo **Address space** identifica se l'indirizzo appartiene a uno spazio riservato (Private Use per RFC 1918, Loopback, Link-local, ecc.) o a Internet pubblico.

Si può fornire la subnet mask anche in forma dotted-decimal invece del prefisso CIDR:

```
ipcalc-ng 192.168.1.47 255.255.255.0
```

Calcolo del numero di host disponibili

Il numero di host utilizzabili in una subnet si calcola come $2^{(32-\text{prefix})} - 2$, sottraendo il network address e il broadcast address:

Prefisso	Subnet mask	Host disponibili	
/24	255.255.255.0	$2^8 - 2 = 254$	
/25	255.255.255.128	$2^7 - 2 = 126$	
/26	255.255.255.192	$2^6 - 2 = 62$	
/27	255.255.255.224	$2^5 - 2 = 30$	
/30	255.255.255.252	$2^2 - 2 = 2$	(link point-to-point)
/32	255.255.255.255	$2^0 - 2 = 0$	(singolo host)

Output esteso: `--all-info`

L'opzione `-all-info` aggiunge rispetto all'output base il reverse DNS della subnet e la classe di appartenenza storica (A, B, C):

```
ipcalc-ng --all-info 192.168.1.47/24
```

```
Address:         192.168.1.47
Network:         192.168.1.0/24
Netmask:         255.255.255.0 = 24
Broadcast:       192.168.1.255
Reverse DNS:     1.168.192.in-addr.arpa.

Address space:   Private Use
Address class:   Class C
HostMin:         192.168.1.1
HostMax:         192.168.1.254
Hosts/Net:       254
```

Su IP pubblici con GeoIP disponibile, `-all-info` aggiunge anche paese e coordinate geografiche:

```
ipcalc-ng --all-info 8.8.8.8
```

```
Address:         8.8.8.8
Reverse DNS:     8.8.8.8.in-addr.arpa.
Address space:   Internet
Address class:   Class A
```

▷ GeoIP

Le informazioni geografiche richiedono che la libreria `libGeoIP` sia installata sul sistema. Se assente, `ipcalc-ng` mostra comunque le informazioni di base senza errori.

Subnetting: `-S` (split)

L'opzione `-S` divide una rete in sottoreti di dimensione specificata. È lo strumento principale per fare subnetting amministrativo.

```
ipcalc-ng -S 26 192.168.1.0/24
```

```
[Split networks]
Network:         192.168.1.0/26
Network:         192.168.1.64/26
Network:         192.168.1.128/26
Network:         192.168.1.192/26

Total:           4
Hosts/Net:       62
```

Il comando divide `192.168.1.0/24` in 4 subnet da `/26`, ciascuna con 62 host utili. Il prefisso passato a `-S` deve essere più grande (subnet più piccole) di quello della rete di partenza.

Esempio pratico: suddivisione per dipartimenti

Supponiamo che l'università abbia ricevuto il blocco `150.146.0.0/16` e voglia assegnare una `/24` a ciascun dipartimento:

```
ipcalc-ng -S 24 150.146.0.0/16
```

Produce 256 subnet da `/24`: `150.146.0.0/24`, `150.146.1.0/24`, ..., `150.146.255.0/24`. Ciascuna con 254 host disponibili, tutte con indirizzi **pubblici**.

> ▷ **Split e indirizzi pubblici**
>
> Il subnetting con `-S` funziona identicamente su blocchi pubblici e privati. Storicamente, prima della RFC 1918 (1996), le università facevano esattamente questo: suddividevano autonomamente il loro blocco pubblico di classe B in tante `/24` interne, senza chiedere nulla alla IANA.

Deaggregazione di range: `-d`

L'opzione `-d` fa l'operazione inversa della route aggregation: dato un range arbitrario di indirizzi, trova i prefissi CIDR minimi che lo coprono esattamente.

```
ipcalc-ng -d 192.168.1.3-192.168.1.67
```

```
[Deaggregated networks]
Network:         192.168.1.3/32
Network:         192.168.1.4/30
Network:         192.168.1.8/29
Network:         192.168.1.16/28
Network:         192.168.1.32/27
Network:         192.168.1.64/30
```

Il range `192.168.1.3`–`192.168.1.67` non corrisponde a nessun singolo prefisso CIDR (non inizia e non finisce su boundary di potenza di 2). `ipcalc-ng` lo decompone nel minor numero di prefissi CIDR che lo coprono esattamente, senza sovrapposizioni e senza buchi.

> ▷ **Uso pratico**
>
> La deaggregazione è utile quando si scrivono **ACL** (Access Control List) o regole firewall: invece di una regola generica su una subnet grande, si ottengono le subnet minime che coprono esattamente gli indirizzi da filtrare, minimizzando il traffico non voluto che rientrerebbe in una subnet più larga.

Output JSON per script: `-j`

L'opzione `-j` produce l'output in formato JSON, adatto per essere processato da script o da altri programmi:

```
ipcalc-ng -j 192.168.1.47/24
```

```
{
   "ADDRESS":     "192.168.1.47",
   "NETWORK":     "192.168.1.0",
```

```
  "NETMASK":    "255.255.255.0",
  "PREFIX":     "24",
  "BROADCAST":  "192.168.1.255",
  "ADDRSPACE":  "Private Use",
  "MINADDR":    "192.168.1.1",
  "MAXADDR":    "192.168.1.254",
  "ADDRESSES":  "254"
}
```

In uno script Python si può usare direttamente:

```python
import subprocess, json

out = subprocess.check_output(
    ["ipcalc-ng", "-j", "192.168.1.47/24"])
data = json.loads(out)
print(data["NETWORK"], data["BROADCAST"])
# 192.168.1.0  192.168.1.255
```

Opzioni singole per script Bash

Quando serve estrarre un solo campo, le opzioni singole producono output in formato `VAR=valore`, direttamente usabile in Bash con `eval`:

```
ipcalc-ng -p 192.168.1.47/24      # PREFIX=24
ipcalc-ng -n 192.168.1.47/24      # NETWORK=192.168.1.0
ipcalc-ng -b 192.168.1.47/24      # BROADCAST=192.168.1.255
ipcalc-ng --minaddr 192.168.1.0/24  # MINADDR=192.168.1.1
ipcalc-ng --maxaddr 192.168.1.0/24  # MAXADDR=192.168.1.254
```

Esempio pratico in uno script Bash:

```
#!/bin/bash
# Estrai network e broadcast in una sola chiamata
eval $(ipcalc-ng -n -b 192.168.1.47/24)
echo "Network:   $NETWORK"
echo "Broadcast: $BROADCAST"

# Verifica se un indirizzo e' in una certa subnet
NET=$(ipcalc-ng -n --no-decorate 192.168.1.47/24)
echo "Appartiene alla rete: $NET"
```

Opzione	Campo restituito
-p	PREFIX
-n	NETWORK
-b	BROADCAST
-m	NETMASK
-minaddr	MINADDR
-maxaddr	MAXADDR
-addrspace	ADDRSPACE

Validazione di indirizzi: `-c`

L'opzione `-c` verifica se un indirizzo è sintatticamente valido. Restituisce exit code `0` se valido, `1` se invalido — ideale per validare input in script:

```
ipcalc-ng -c 192.168.1.47     # exit code 0 -- valido
ipcalc-ng -c 300.168.1.47     # exit code 1 -- invalido
# ipcalc-ng: bad IPv4 address: 300.168.1.47

ipcalc-ng -c -4 192.168.1.47   # forza IPv4
ipcalc-ng -c -6 ::1            # forza IPv6
```

Utilizzo in uno script Bash:

```
#!/bin/bash
read -p "Inserisci un indirizzo IP: " addr
if ipcalc-ng -c "$addr" 2>/dev/null; then
    echo "Indirizzo valido: $addr"
else
    echo "Indirizzo non valido." >&2
    exit 1
fi
```

Supporto IPv6

`ipcalc-ng` supporta IPv6 con la stessa sintassi di IPv4. È uno dei principali vantaggi rispetto al classico `ipcalc`:

```
ipcalc-ng 2001:db8::/32
```

```
Full Network:   2001:0db8:0000:0000:0000:0000:0000:0000/32
Network:        2001:db8::/32
Netmask:        ffff:ffff:: = 32

Address space:  Documentation
HostMin:        2001:db8::
HostMax:        2001:db8:ffff:ffff:ffff:ffff:ffff:ffff
Hosts/Net:      2^(96) = 79228162514264337593543950336
```

Il campo **Full Network** mostra l'indirizzo espanso senza abbreviazioni, utile per capire la struttura completa dell'indirizzo. Il campo **Address space** riconosce automaticamente i prefissi speciali IPv6:

Prefisso	Address space	Uso
`::1/128`	Loopback	Loopback locale
`fe80::/10`	Link-scoped	Link-local, obbligatorio
`fc00::/7`	Unique Local	Privato IPv6 (come RFC 1918)
`2001:db8::/32`	Documentation	Esempi e documentazione
`2000::/3`	Global Unicast	Indirizzi pubblici

Lo split funziona anche su IPv6:

```
ipcalc-ng -S 64 2001:db8::/32
# Divide il /32 in subnet /64
# Ogni /64 contiene 2^64 host
```

Confronto tra versioni

Funzionalità	ipcalc classico	ipcalc-ng	sipcalc
Analisi subnet IPv4	✓	✓	✓
Supporto IPv6	✗	✓	✓
Output JSON	✗	✓	✗
Split subnet (`-S`)	limitato	✓	✓
Deaggregazione range	✓	✓	✗
GeoIP	✗	✓	✗
Output hex / decimal	✗	✗	✓
Cisco wildcard mask	✓	✗	✓
Manutenzione attiva	✗	✓	✗

▷ Quale usare

Per uso didattico e per script moderni: `ipcalc-ng`. Per visualizzare la rappresentazione esadecimale o la Cisco wildcard mask: `sipcalc`. Il classico `ipcalc` Red Hat va bene per calcoli rapidi su sistemi enterprise dove è già installato, ma non installarlo ex novo su sistemi nuovi.

Riepilogo dei comandi principali

Comando	Scopo
`ipcalc-ng 192.168.1.47/24`	Analisi base della subnet
`ipcalc-ng -all-info 192.168.1.47/24`	Analisi estesa (reverse DNS, classe)
`ipcalc-ng -S 26 192.168.1.0/24`	Split: divide /24 in subnet /26
`ipcalc-ng -d 192.168.1.3-192.168.1.67`	Deaggregate: range → prefissi CIDR
`ipcalc-ng -j 192.168.1.47/24`	Output JSON per script
`ipcalc-ng -p 192.168.1.47/24`	Solo PREFIX=24
`ipcalc-ng -n 192.168.1.47/24`	Solo NETWORK=192.168.1.0
`ipcalc-ng -b 192.168.1.47/24`	Solo BROADCAST=192.168.1.255
`ipcalc-ng -c 192.168.1.47`	Valida indirizzo (exit code 0/1)
`ipcalc-ng 2001:db8::/32`	Analisi subnet IPv6
`ipcalc-ng -all-info 8.8.8.8`	Info + GeoIP su IP pubblico

Consegna: inviare zip con report e sorgenti al docente (`roberto.marino@unime.it`)
Riferimento: `man ipcalc-ng` — `ipcalc-ng -help` — `https://gitlab.com/ipcalc/ipcalc`