

# Practical Uses of nc(netcat) command in Linux

Last Updated : 08 Jun, 2023

**Netcat** is one of the most powerful **networking tools, security tools, and network monitoring tools**. It acts like a [cat command](#) over a network. It is even considered a Swiss army knife of networking tools. It is generally used for the following reasons:

- Operation related to TCP, UDP, or UNIX-domain sockets.
- Port Scanning
- Port Listening
- Port redirection
- open Remote connections
- Read/Write data across the network.
- Network debugging
- Network daemon testing
- Simple TCP proxies
- A Socks or HTTP Proxy Command for ssh

It is designed by keeping in mind that it should be a flexible “back-end” tool that can be used directly or driven by any other program.

## Installing Netcat (nc) Process Monitoring Tool in Linux

To install the Netcat tool use the following commands as per your Linux distribution.

### In the case of Debian/Ubuntu

```
sudo apt-get install netcat
```

### In the case of CentOS/RHEL

```
sudo yum install nc
```

### In the case of Fedora 22+ and RHEL 8,9

```
sudo dnf install nc
```

**NOTE:** To verify that it is successfully installed in our system, we run the following command “nc -h”. This will display the help menu of Netcat, indicating that it is installed and ready to be used.

## Syntax of the `nc` command in Linux

The basic syntax for the nc command as follows.

```
nc [options] [hostname] [port]
```

**[options]:** Use to customize the behavior of the nc command. Some examples include adding verbose output, setting a timeout for connections, etc.

**[hostname]:** The hostname or the IP address of the target we want to connect to or interact with using netcat. It can be both Domain name (e.g., example.com) or an IP address (e.g., 192.168.0.1).

**[port]:** The port number of the target that we want to connect to or interact with. Ports are used to identify specific services or applications running on a system. (For example: port 80 (HTTP) or port 22 (SSH) connections.

## netcat (nc) command options

It offers various options that help us in enhancing its functionality. Some commonly used options include:

Options	Description
<b>-l</b>	Listen mode, used to create a server that listens for incoming connections.
<b>-p</b>	Specifies the source port number.
<b>-v</b>	Verbose mode, provides more detailed output.
<b>-z</b>	Scans for open ports.
<b>-w</b>	Sets a timeout for connections.
<b>-q</b>	Specifies the delay before closing the connection.

## Two Primary Working Modes of Netcat.

We have two primary working modes:

### Connect Mode

In this mode Netcat works as a client. Which means that it establishes a connection to a remote server or servers. To work in this mode, we have to provide the `**<host>**` and `**<port>**` parameters.

**<host>** = In this we have to provide the hostname or IP address of the remote server or servers. It can be both Domain name (e.g., example.com) or an IP address (e.g., 192.168.0.1).

**<port>** = In this we specify the port number on the remote server or services that we want to connect to. Basically, it represents the endpoint where the desired service is running.

**For example:** If we want to establish a connection to the HTTP (web) service running on domain name (example.com) or IP address (e.g., 192.168.0.1) on port 80. We use the following command.

```
nc example.com 80
```

### **Listen Mode**

In this mode Netcat works as a server. Which means that it waits and listens for incoming connections from clients. To work in this mode, we have to use Netcat in Listen mode and provide the `**<host>**` (optional) and `**<port>**` parameters.

**<host>** = it is optional to provide host name, if we do so the Netcat listens on the specific host for incoming connections on the specified `**<port>**`. We can say that it will bind itself to the specified host's IP address or network interface.

**<port>** = In this we specify the port number on which Netcat should listen for incoming connection.

**For example:** If we want to listen to the IP address (e.g., 192.168.0.1) on port 8080. We use the following command.

```
nc -l 192.168.0.1 8080
```

We can use `**-lv**` option to view verbose (-v)

## **Practical implementation of Netcat Security Tool in Linux**

We have two systems running on same network, To know there IP address:

**System 1 IP address (Localhost) (10.143.90.24)**

```
ifconfig
```



*ifconfig*

**System 2 IP address (gfgubun1) (10.143.90.106)**

**ifconfig**



*ifconfig*

## **Client and Server Connection with Netcat (nc) in Linux.**

Here our system 1 will act as a server (listens) and system 2 will act as a client (connects)

### **1) System 1**

We are running `nc` command in listen mode and providing a port number.

```
nc -lv 1111
```

You can replace port number `1111` with your desired port number.

*nc -lv 1111*

Here we have used `-l` to make our system 1 a server and used `-v` for verbose to see if it was successful or not.

## 2) System 2

We are running `nc` command with IP address of System 1 and a port number on which system 1 is listening.

```
nc -v 10.143.90.24 1111
```

*nc -v 10.143.90.24 1111*



used `-v` for verbose to see if it was successful or not And mentioned IP address of System1.

IF we send message from any System we will see that message on both the systems.

## **Scanning Ports with netcat (nc) in Linux**

### **1) Making System 2 to listen on port 1111.**

```
nc -lvk 1111
```

*nc -lvk 1111*

used `-v` for verbose to see if it was successful or not and used `-k` so that our connection doesn't stop in case of disconnect.

### **2) Checking If the port is open From System 1.**

Here we will check for port number 1111, if this port is open, we will see successful in connection. (used `-v` for verbose to see if it was succeeded)

```
nc -zv 10.143.90.106 1111
```

Here we have used `-z` option, to scan for open ports.

### **3) Finding Open Ports in a range of ports**

IF we want to search in between the range of ports that are Open, we can write a small script as follows.

```
vim port_scan.sh
```

You can replace file name port\_scan.sh with your requirements.

Here You have to replace the "10.143.90.106" with you requirement,  
We are taking start port and end port as a input from user itself.

```
#!/bin/bash
```

```
host="10.143.90.106"
```

```
read -p "Enter the starting port number: " start_port
```

```
read -p "Enter the ending port number: " end_port
```

```
for (( port=start_port; port<=end_port; port++ ))
```

```
do
```

```
  nc -zv "$host" "$port"
```

```
done
```

*port\_scan.sh*

Making our script executable.

```
chmod +x port_scan.sh
```

Running script.

```
./port_scan.sh
```

Then enter the starting port and ending port.

*1111 port is open*

if this port is open, we will see success in connection. Here we can see port `1111` is open.

**Transfer File using Netcat (nc) in Linux.**

If we have a file name “file\_1.txt” in our system 2 and want to transfer that file in system 1, we use the following command.

**In system 2**

```
nc -lv 10.143.90.106 1111 < file_1.txt
```

**In system 1**

```
nc -zv 10.143.90.106 1111 > file_1.txt
```

Then we use `ls` command to confirm that we have received file in our system 1.

*file\_1.txt transfer successfully in our system 1*



## Chat Server using Netcat (nc) in Linux.

On system 2





**1.** To start listening on a port, first Open 2 terminal windows. *Terminal 1 for listening*

```
$nc -l -p 1234
```

*Terminal 2 sending request*

```
$nc 127.0.0.1 1234
```

**Note:** Here the port number is 1234 and by default host is the localhost.

```
manav@manav-VirtualBox:~$ nc 127.0.0.1 1234
```

```
manav@manav-VirtualBox: ~
```

```
manav@manav-VirtualBox:~$ nc -l -p 1234
```

It will not display anything but will start listening to port 1234 at the localhost from terminal 1. And anything entered in terminal 2 will be reflected back in terminal 1 as well which confirms that the connection is established successfully.

2. To transfer data. Open 2 terminal windows. *Terminal 1 for listening*

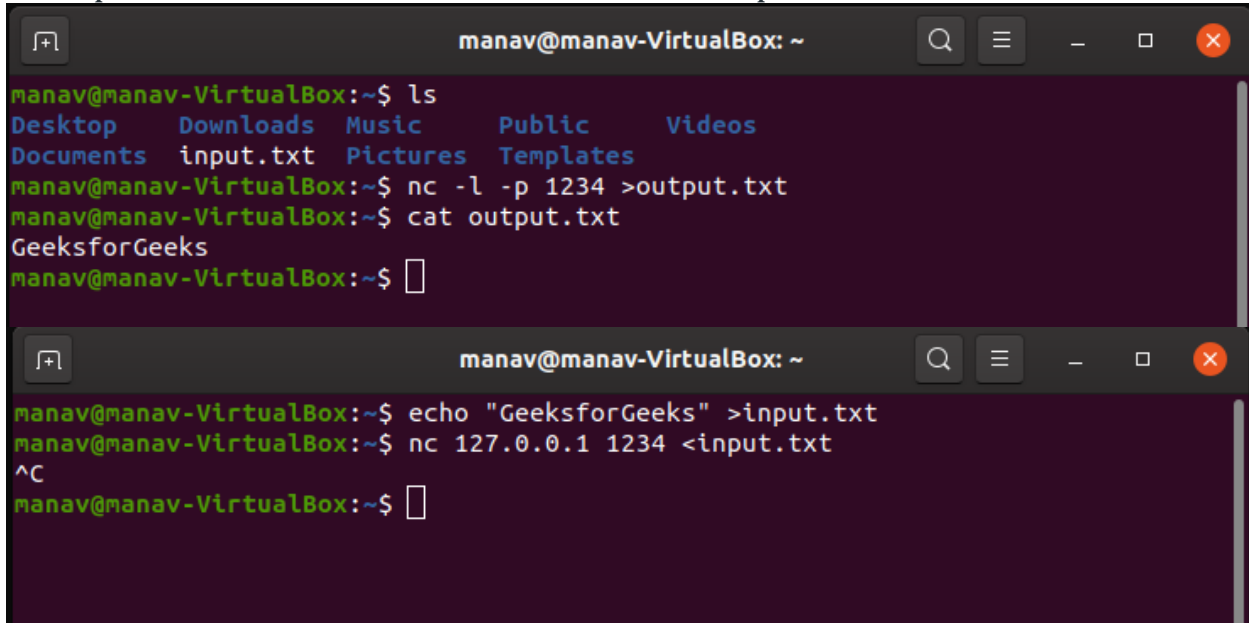
```
$nc -l -p 1234 >output.txt
```

*Terminal 2 for sending request*

```
$echo "GeeksforGeeks" >input.txt
```

```
$nc 127.0.0.1 1234 <input.txt
```

**Note:** Here the port number is 1234 and by default host is localhost. It will send the input.txt file's data from terminal 2 to the output.txt file at terminal 1.



```
manav@manav-VirtualBox: ~  
manav@manav-VirtualBox:~$ ls  
Desktop    Downloads  Music      Public     Videos  
Documents  input.txt  Pictures   Templates  
manav@manav-VirtualBox:~$ nc -l -p 1234 >output.txt  
manav@manav-VirtualBox:~$ cat output.txt  
GeeksforGeeks  
manav@manav-VirtualBox:~$  
  
manav@manav-VirtualBox:~$ echo "GeeksforGeeks" >input.txt  
manav@manav-VirtualBox:~$ nc 127.0.0.1 1234 <input.txt  
^C  
manav@manav-VirtualBox:~$
```

3. To perform Port Scanning. Enter the following command on the terminal. *Scanning a single port*

```
$netcat -z -v 127.0.0.1 1234
```

*Scanning multiple ports*

```
$nc -z -v 127.0.0.1 1234 1235
```

*Scanning a range of ports*

```
$nc -z -v 127.0.0.1 1233-1240
```

**Note:** Here the port numbers are 1234, 1235, 1233, and 1240 you may change them as per your need. It will display the port number with the status(open or not).

```
manav@manav-VirtualBox: ~  
manav@manav-VirtualBox:~$ nc -z -v 127.0.0.1 1234  
Connection to 127.0.0.1 1234 port [tcp/*] succeeded!  
manav@manav-VirtualBox:~$ nc -z -v 127.0.0.1 1234 1235  
Connection to 127.0.0.1 1234 port [tcp/*] succeeded!  
Connection to 127.0.0.1 1235 port [tcp/*] succeeded!  
manav@manav-VirtualBox:~$ nc -z -v 127.0.0.1 1233-1240  
nc: connect to 127.0.0.1 port 1233 (tcp) failed: Connection refused  
Connection to 127.0.0.1 1234 port [tcp/*] succeeded!  
Connection to 127.0.0.1 1235 port [tcp/*] succeeded!  
nc: connect to 127.0.0.1 port 1236 (tcp) failed: Connection refused  
Connection to 127.0.0.1 1237 port [tcp/*] succeeded!  
nc: connect to 127.0.0.1 port 1238 (tcp) failed: Connection refused  
nc: connect to 127.0.0.1 port 1239 (tcp) failed: Connection refused  
nc: connect to 127.0.0.1 port 1240 (tcp) failed: Connection refused  
manav@manav-VirtualBox:~$
```

#### 4. To send an HTTP Request

```
$printf "GET /nc.1 HTTPs/1.1\r\nHost: www.geeksforgeeks.org\r\n\r\n" | nc  
www.geeksforgeeks.org 80
```

**Note:** Here the website is [www.geeksforgeeks.org](http://www.geeksforgeeks.org), you may choose any. It will send a HTTP Request to [www.geeksforgeeks.org](http://www.geeksforgeeks.org).

```
manav@manav-VirtualBox: ~  
manav@manav-VirtualBox:~$ printf "GET /nc.1 HTTPs/1.1\r\nHost: www.geeksforgeeks  
.org\r\n\r\n" | nc www.geeksforgeeks.org 80  
HTTP/1.0 400 Bad Request  
Server: AkamaiGHost  
Mime-Version: 1.0  
Content-Type: text/html  
Content-Length: 216  
Expires: Sat, 25 Apr 2020 00:12:15 GMT  
Date: Sat, 25 Apr 2020 00:12:15 GMT  
Connection: close  
  
<HTML><HEAD>  
<TITLE>Bad Request</TITLE>  
</HEAD><BODY>  
<H1>Bad Request</H1>  
Your browser sent a request that this server could not understand.<P>  
Reference&#32;&#35;7&#46;2daef82d&#46;1587773535&#46;0  
</BODY>  
</HTML>  
manav@manav-VirtualBox:~$
```

#### 5. To delay the interval for lines sent. Open 2 terminal as shown below: *Terminal 1 for listening*

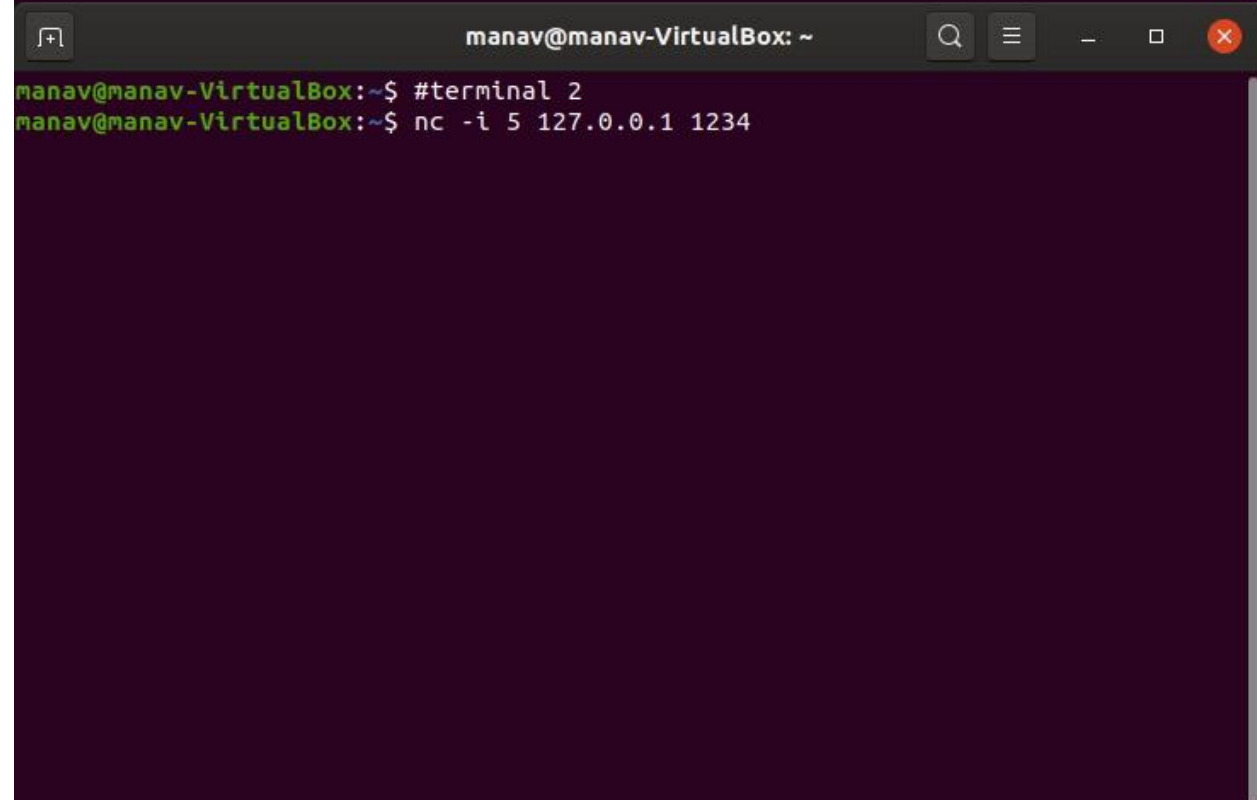
```
$nc -l -p 1234
```

*Terminal 2 sending request*

```
$nc -i 5 127.0.0.1 1234
```

**Note:** Here the port number is 1234 and by default host is localhost. The time taken is 5 seconds. Each will be sent after 5 seconds of time.

```
manav@manav-VirtualBox:~$ #terminal 1  
manav@manav-VirtualBox:~$ nc -l 1234
```



```
manav@manav-VirtualBox: ~  
manav@manav-VirtualBox:~$ #terminal 2  
manav@manav-VirtualBox:~$ nc -i 5 127.0.0.1 1234
```