Penetration Testing Lab Setup: Microsocks

November 16, 2018 By Raj Chandel

Hello friends!! In our previous article we have discussed "Web Proxy Penetration Lab Setup Testing using Squid" and today's article we are going to set up SOCKS Proxy to use it as a Proxy Server on Ubuntu/Debian machines and will try to penetrate it.

Table of Content

Introduction to Proxy

A proxy is a computer system or program that acts as a kind of middle-man or an intermediary to come between your web browser and another computer. Your ISP operates servers– computers designed to deliver information to other computers. It uses proxy servers to accelerate the transfer of information between the server and your computer.

For Example, Two users say A and B both have requested to access the same website of the server then Instead of retrieving the data from the original server, the proxy has "stored or cached" a copy of that site and sends it to User A without troubling the main server.

What is SOCKS Proxy?

A SOCKS server is an all-purpose proxy server that creates a TCP connection to another server on the client's behalf, then exchanges network packets between a client and server. The Tor onion proxy software serves a SOCKS interface to its clients. Even SSH tunnel makes all the connections as per the SOCKS protocol.

For high security, you can go with SOCKS5 protocol that provides various authentication options which you cannot get with the SOCKS4 protocol.

Difference Between Socks proxy and HTTP Proxy

- SOCKS Proxy is low-level which is designed to be a general proxy that will be able to accommodate effectively any protocol, program, or type of traffic.

- SOCKS proxies support both TCP and UDP transfer protocols
- SOCKS performs at Layer 5 of the OSI model SOCKS server
- Accepts an incoming client connection on TCP port 1080.
- HTTP proxies proxy HTTP requests, while SOCKS proxies proxy socket connections
- HTTP proxies are High-Level which are designed for a specific protocol.
- HTTP proxies can only process requests from applications that use the HTTP protocol.
- An HTTP proxy is for proxying HTTP or web traffic at layer 7
- Accepts an incoming client connection on HTTP port 3128.

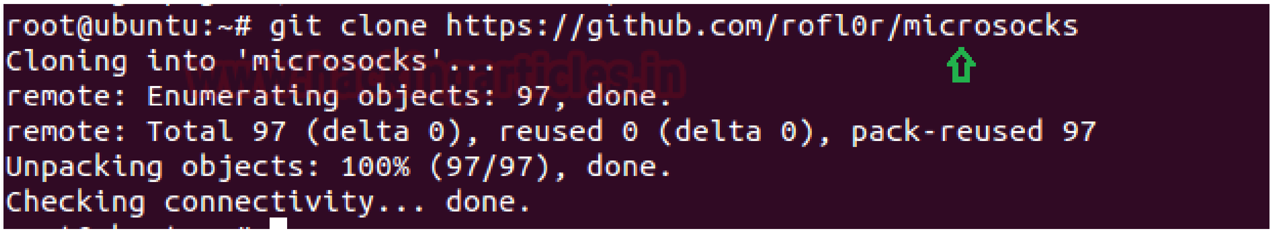
Socks Proxy Installation

For socks proxy lab set-up we are going to download microsocks through GitHub. MicroSocks is multithreaded, small, efficient SOCKS5 server. It's very lightweight, and very light on resources too. Even for every client, a thread with a stack size of 8KB is spawned.

Let's start!!

Open the terminal with sudo rights and enter the following command:

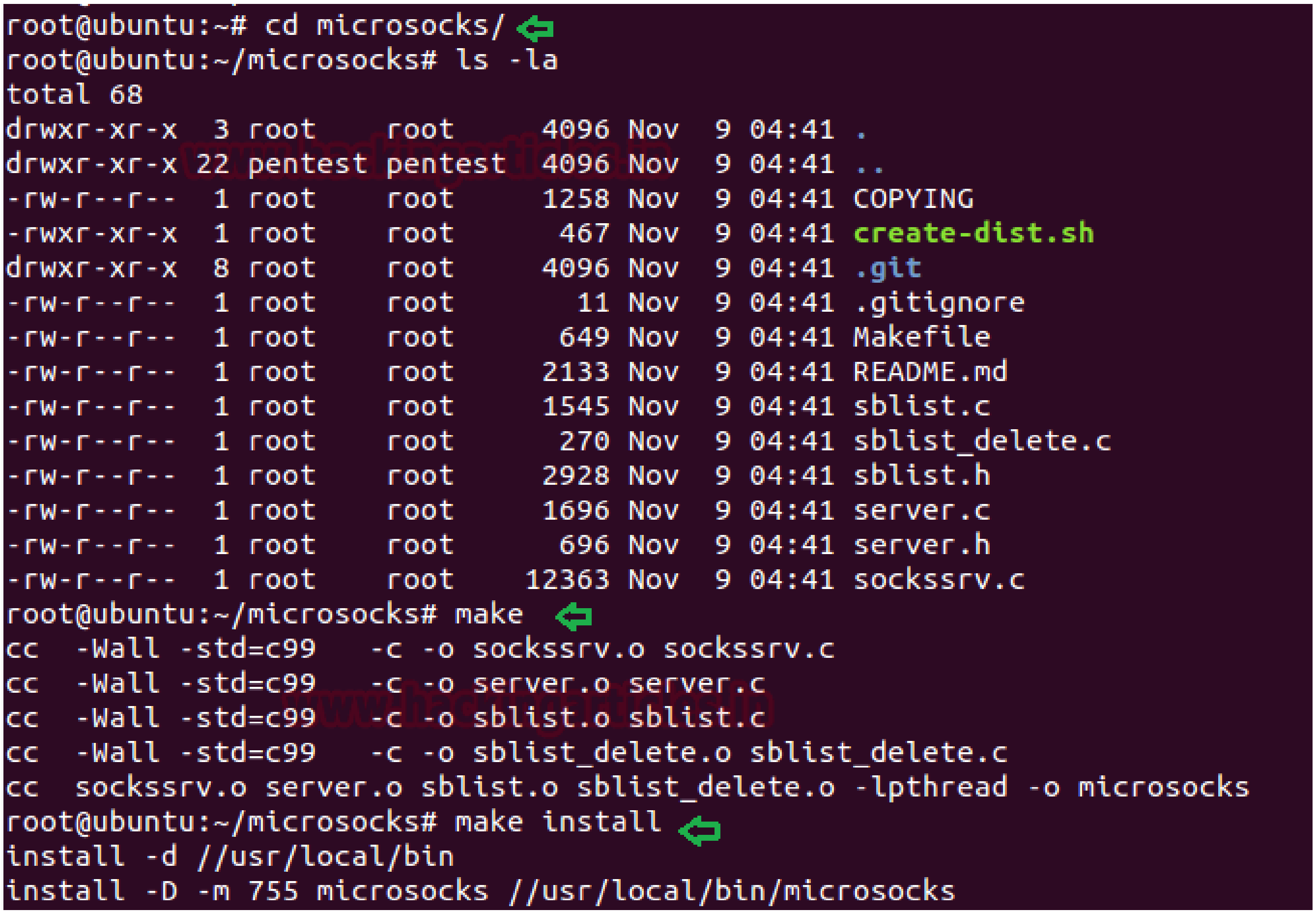
```
git clone https://github.com/rofl0r/microsocks.git
```



```
root@ubuntu:~# git clone https://github.com/rofl0r/microsocks
Cloning into 'microsocks'...
remote: Enumerating objects: 97, done.
remote: Total 97 (delta 0), reused 0 (delta 0), pack-reused 97
Unpacking objects: 100% (97/97), done.
Checking connectivity... done.
```

Once downloading is completed run the following command for its installation:

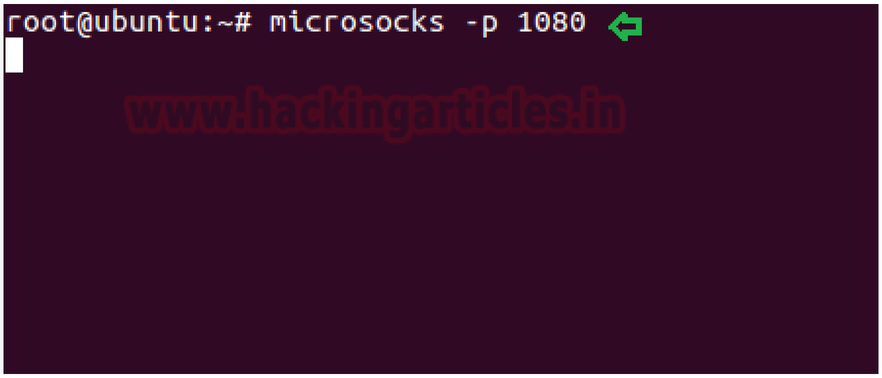
```
cd microsocks
apt install gcc
make
make install
```



```
root@ubuntu:~# cd microsocks/
root@ubuntu:~/microsocks# ls -la
total 68
drwxr-xr-x  3 root    root     4096 Nov  9 04:41 .
drwxr-xr-x 22 pentest pentest  4096 Nov  9 04:41 ..
-rw-r--r--  1 root    root     1258 Nov  9 04:41 COPYING
-rwxr-xr-x  1 root    root      467 Nov  9 04:41 create-dist.sh
drwxr-xr-x  8 root    root     4096 Nov  9 04:41 .git
-rw-r--r--  1 root    root       11 Nov  9 04:41 .gitignore
-rw-r--r--  1 root    root      649 Nov  9 04:41 Makefile
-rw-r--r--  1 root    root     2133 Nov  9 04:41 README.md
-rw-r--r--  1 root    root     1545 Nov  9 04:41 sblist.c
-rw-r--r--  1 root    root      270 Nov  9 04:41 sblist_delete.c
-rw-r--r--  1 root    root     2928 Nov  9 04:41 sblist.h
-rw-r--r--  1 root    root     1696 Nov  9 04:41 server.c
-rw-r--r--  1 root    root      696 Nov  9 04:41 server.h
-rw-r--r--  1 root    root    12363 Nov  9 04:41 sockssrv.c
root@ubuntu:~/microsocks# make
cc  -Wall -std=c99   -c -o sockssrv.o sockssrv.c
cc  -Wall -std=c99   -c -o server.o server.c
cc  -Wall -std=c99   -c -o sblist.o sblist.c
cc  -Wall -std=c99   -c -o sblist_delete.o sblist_delete.c
cc  sockssrv.o server.o sblist.o sblist_delete.o -lpthread -o microsocks
root@ubuntu:~/microsocks# make install
install -d //usr/local/bin
install -D -m 755 microsocks //usr/local/bin/microsocks
```

Now execute the following command to run socks proxy on port 1080 without authentication.

```
microsocks -p 1080
```



```
root@ubuntu:~# microsocks -p 1080
```

As you can observe FTP, SSH, HTTP and Socks are running in our local machine and now let's go for socks penetration testing on a various protocol to ensure whether it is an all-purpose program or not as said above.

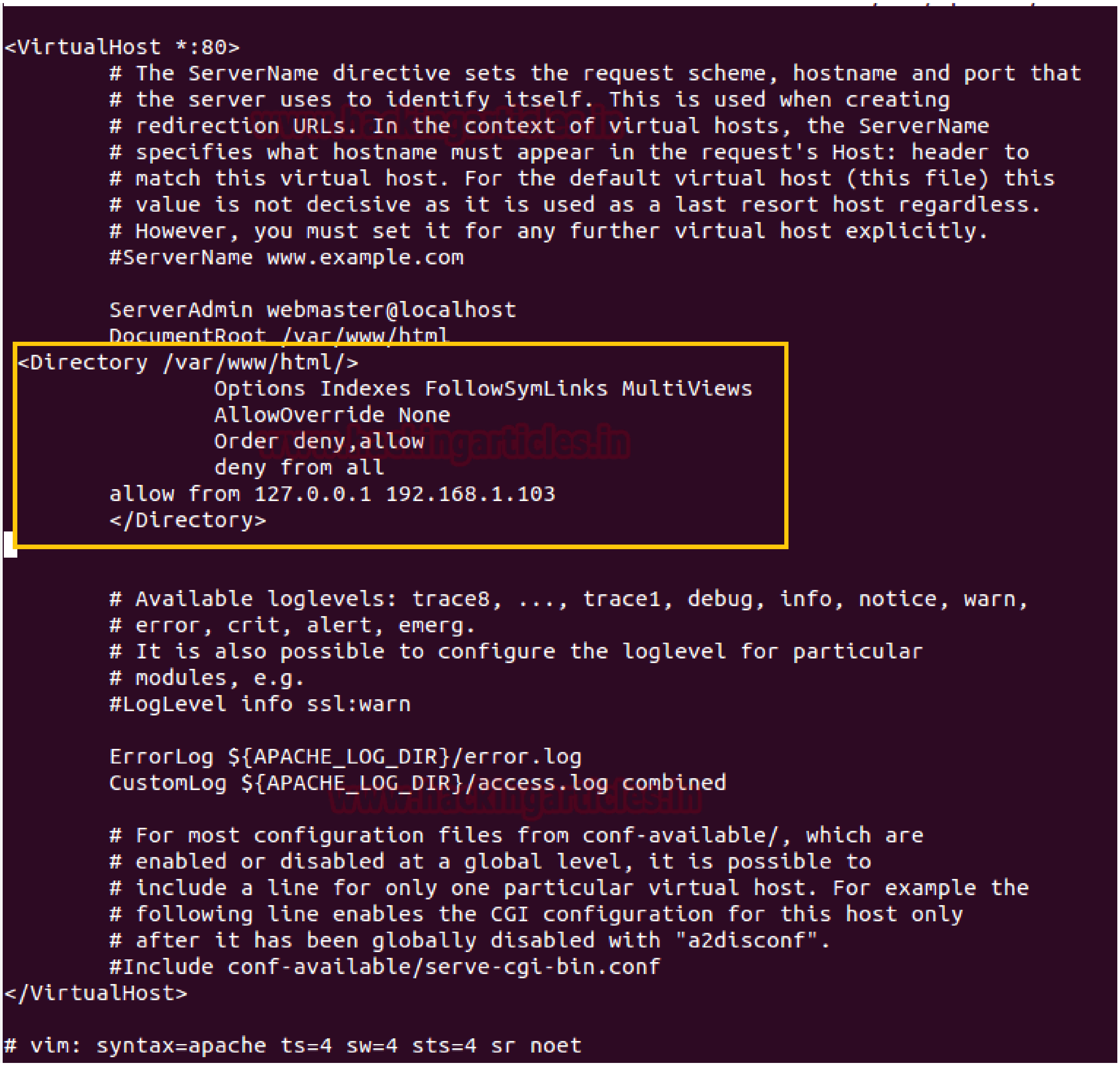
Connecting HTTP via Proxy

Now Configuring Apache service for Web Proxy, therefore, open the "000-default.conf" file from the path: /etc/apache2/sites-available/ and add following line to implement the following rules on /html directory over localhost or Machine IP (192.168.1.103).

```
<Directory /var/www/html/>
                Options Indexes FollowSymLinks MultiViews
                AllowOverride None
                Order deny,allow
                deny from all
        allow from 127.0.0.1 192.168.1.103
</Directory>
```

Now the save the file and restart the apache service with the help of the following command.

```
service apache2 start
```

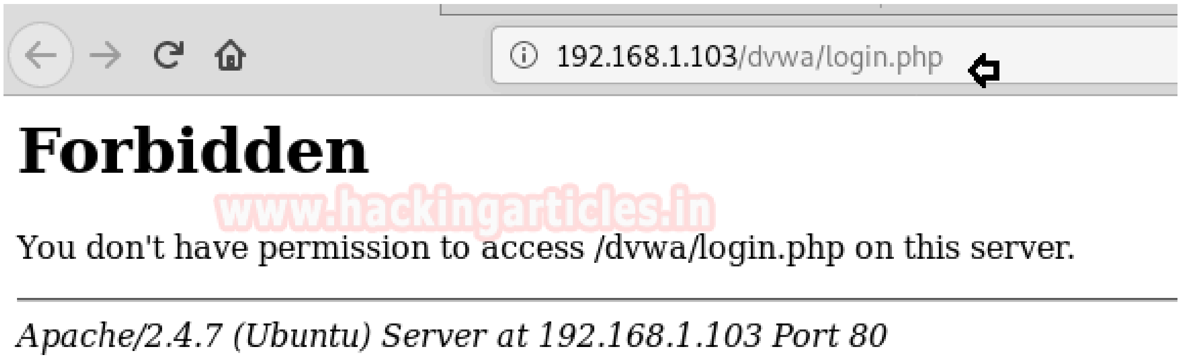


```
<VirtualHost *:80>
        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) this
        # value is not decisive as it is used as a last resort host regardless.
        # However, you must set it for any further virtual host explicitly.
        #ServerName www.example.com

        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html
 <Directory /var/www/html/>
                Options Indexes FollowSymLinks MultiViews
                AllowOverride None
                Order deny,allow
                deny from all
        allow from 127.0.0.1 192.168.1.103
        </Directory>


        # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
        # error, crit, alert, emerg.
        # It is also possible to configure the loglevel for particular
        # modules, e.g.
        #LogLevel info ssl:warn

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

        # For most configuration files from conf-available/, which are
        # enabled or disabled at a global level, it is possible to
        # include a line for only one particular virtual host. For example the
        # following line enables the CGI configuration for this host only
        # after it has been globally disabled with "a2disconf".
        #Include conf-available/serve-cgi-bin.conf
</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Now when someone tries to access web services through our network i.e. 192.168.1.103, he/she will welcome by following web page

“Error 403 forbidden You don’t have permission to access <requested page>”.

When you face that such type of situation where port 80 is open but you are unable to access it, hence proved the network is running behind a proxy server.



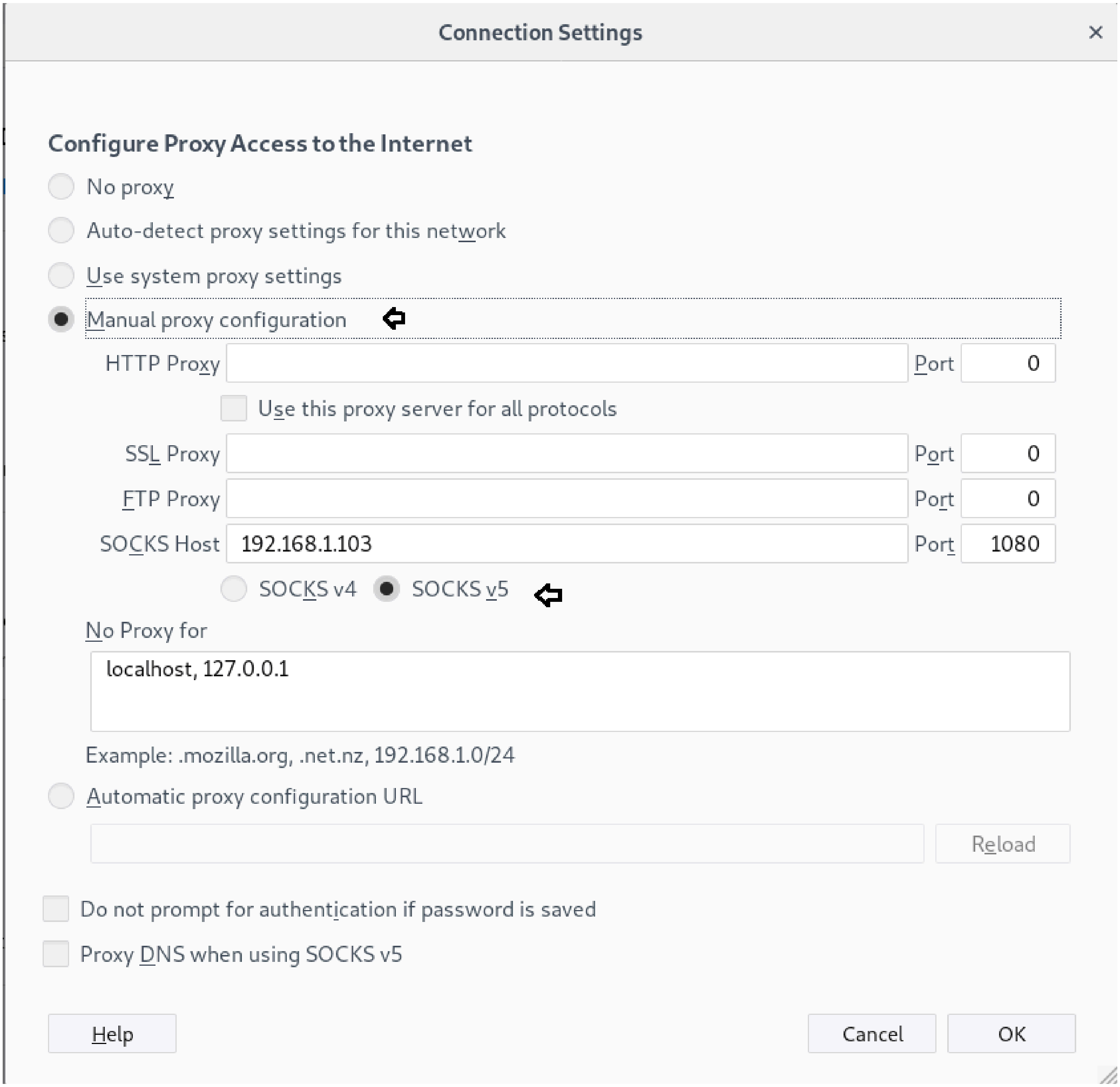
Forbidden

You don't have permission to access /dvwa/login.php on this server.

Apache/2.4.7 (Ubuntu) Server at 192.168.1.103 Port 80

For web Proxy penetration testing we had already set-up lab for web application server such as DVWA (Read Article from here).

Now to test whether our proxy server is working or not by configuring , let's open Firefox and go to **Edit** –> **Preferences** –> **Advanced** –> **Network** –> **Settings** and then select "Manual proxy configuration" and enter SOCKS proxy server IP address (192.168.1.103) and Port (1080) to be used for all protocol.



Connection Settings

Configure Proxy Access to the Internet

○ No proxy

○ Auto-detect proxy settings for this network

○ Use system proxy settings

● Manual proxy configuration

HTTP Proxy: | Port: 0

□ Use this proxy server for all protocols

SSL Proxy: | Port: 0

FTP Proxy: | Port: 0

SOCKS Host: 192.168.1.103 | Port: 1080

○ SOCKS v4 ● SOCKS v5

No Proxy for

localhost, 127.0.0.1

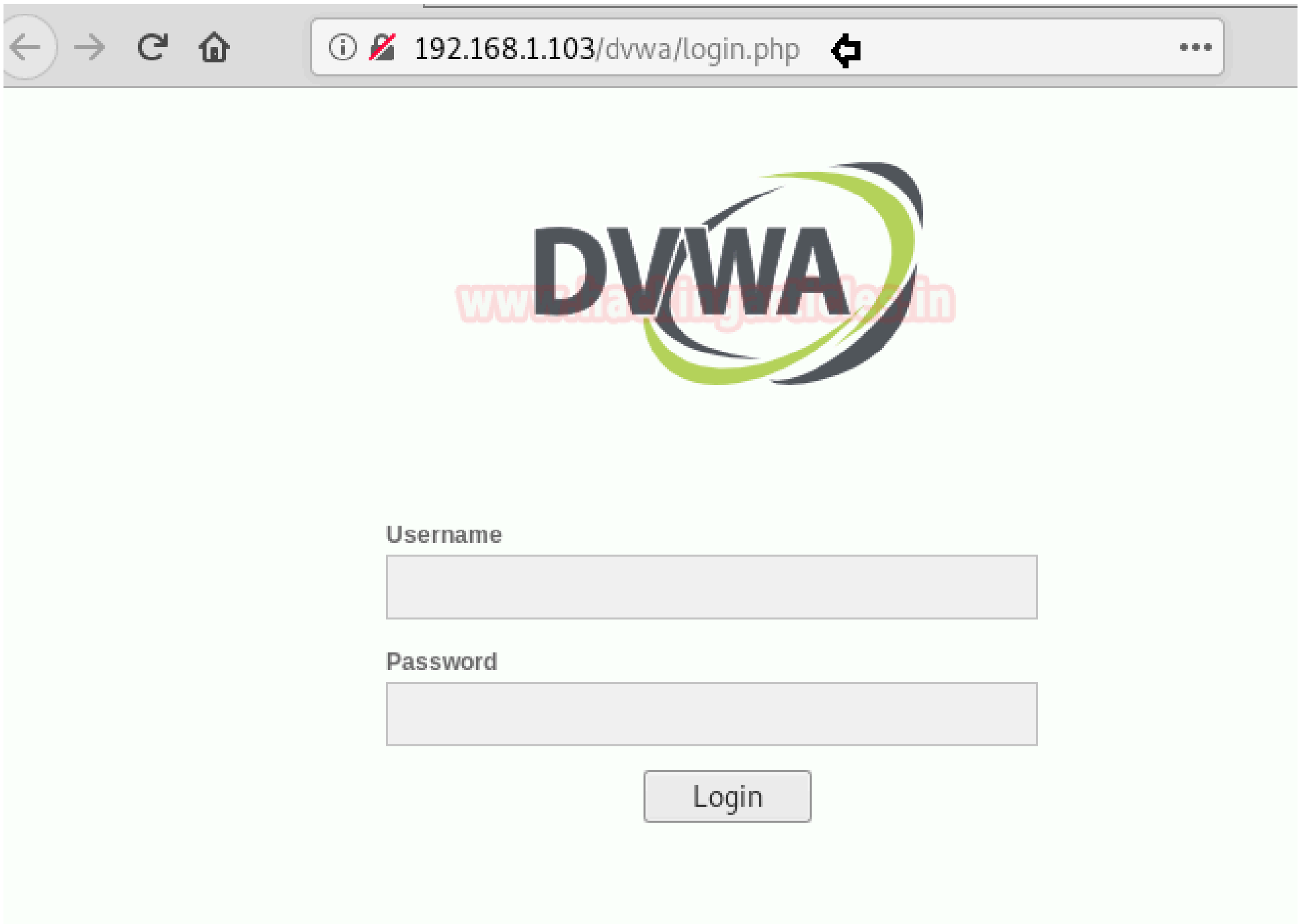
Example: .mozilla.org, .net.nz, 192.168.1.0/24

○ Automatic proxy configuration URL

□ Do not prompt for authentication if password is saved

□ Proxy DNS when using SOCKS v5

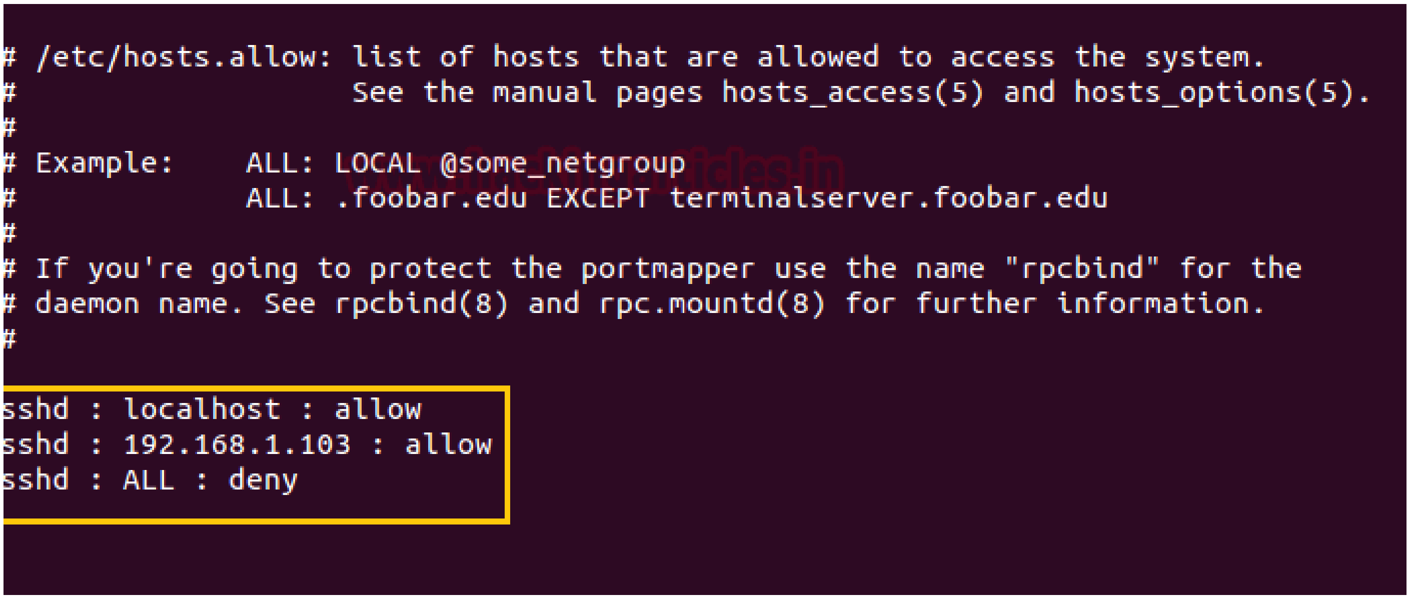
BOOMMM!! Connected to the Proxy server successfully using HTTP Proxy in our Browser.



Connecting SSH via Proxy

Now configuring host.allow file for SSH Proxy, therefore, open /etc/hosts.allow file and following line to allow SSH connection on localhost IP and restrict for others.

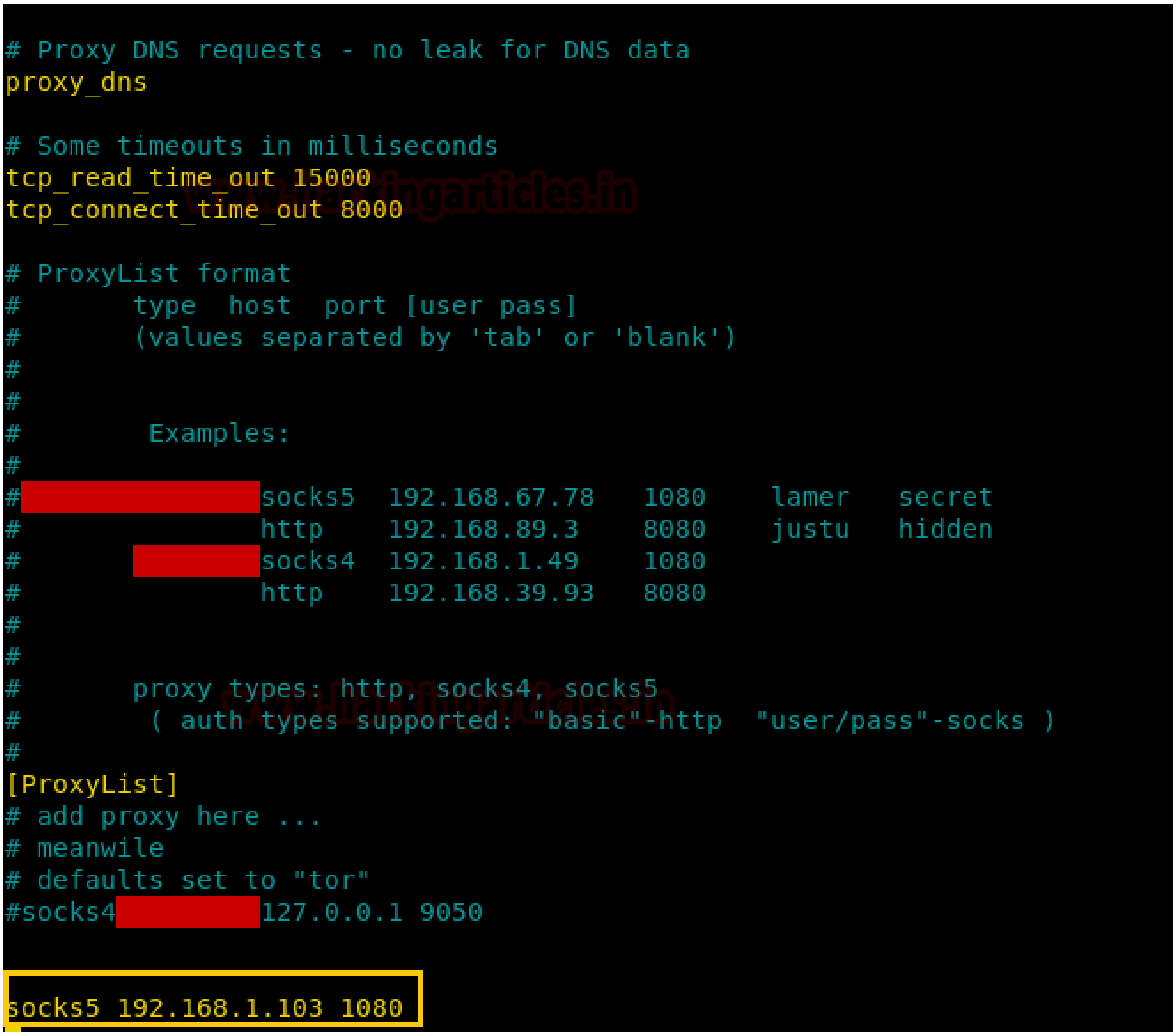
```
sshd : localhost : allow
sshd : 192.168.1.103 : allow
sshd : ALL : deny
```



```
# /etc/hosts.allow: list of hosts that are allowed to access the system.
#                   See the manual pages hosts_access(5) and hosts_options(5).
#
# Example:    ALL: LOCAL @some_netgroup
#             ALL: .foobar.edu EXCEPT terminalserver.foobar.edu
#
# If you're going to protect the portmapper use the name "rpcbind" for the
# daemon name. See rpcbind(8) and rpc.mountd(8) for further information.
#

sshd : localhost : allow
sshd : 192.168.1.103 : allow
sshd : ALL : deny
```

Now open a proxychains configuration file from the given path */etc/proxychains.conf* in your Kali Linux and then add the following line at the bottom.

```
socks5 192.168.1.103 1080
```



```
# Proxy DNS requests - no leak for DNS data
proxy_dns

# Some timeouts in milliseconds
tcp_read_time_out 15000
tcp_connect_time_out 8000

# ProxyList format
#       type  host  port [user pass]
#       (values separated by 'tab' or 'blank')
#
#
#        Examples:
#
#[redacted]socks5  192.168.67.78   1080    lamer   secret
#               http    192.168.89.3    8080    justu   hidden
#       [redacted]socks4  192.168.1.49    1080
#               http    192.168.39.93   8080
#
#
#       proxy types: http, socks4, socks5
#        ( auth types supported: "basic"-http  "user/pass"-socks )
#
[ProxyList]
# add proxy here ...
# meanwile
# defaults set to "tor"
#socks4[redacted]127.0.0.1 9050

socks5 192.168.1.103 1080
```

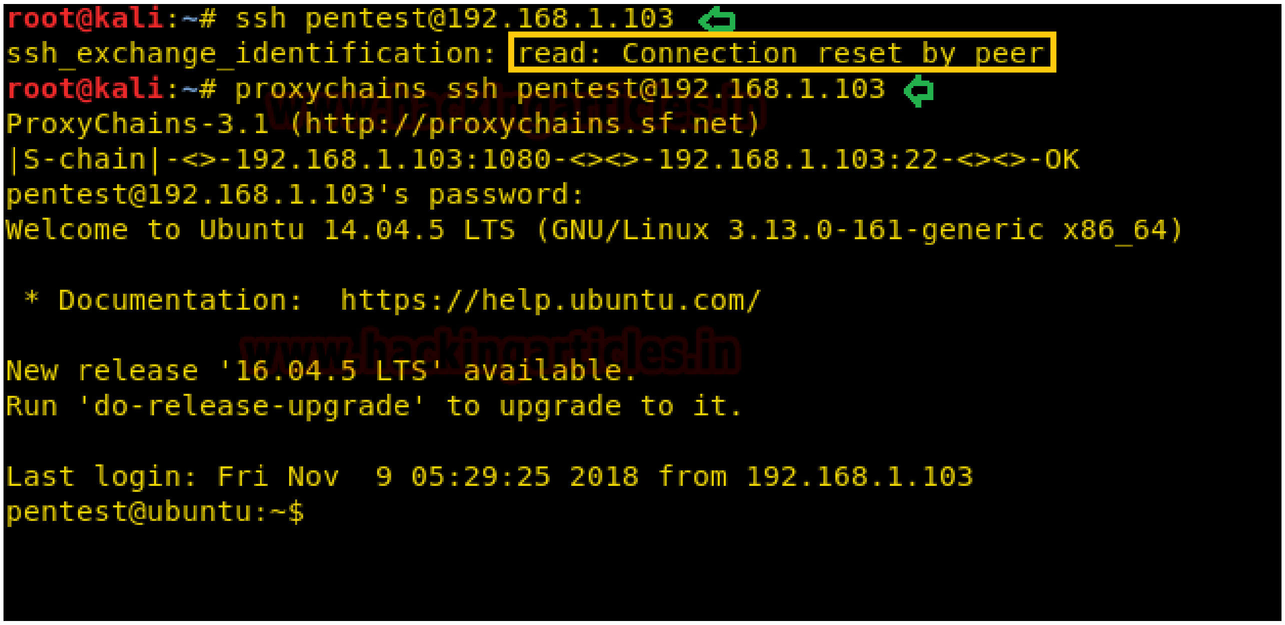
Now when we try to connect with target machine via port 22 for SSH connection we got an error message "Connection reset by peer" as shown in below image after executing the 1st command.

```
ssh pentest@192.168.1.103
```

When you face that such type of situation where port 22 is open but you are unable to access it, hence proved the network is running behind the proxy server.

But if you will use **proxychains** along with the command after saving the configuration as said above then you can easily connect with target network via port 22 for ssh connection as shown in below image after executing the 2nd command.

```
proxychains ssh pentest@192.168.1.103
```



```
root@kali:~# ssh pentest@192.168.1.103
ssh_exchange_identification: read: Connection reset by peer
root@kali:~# proxychains ssh pentest@192.168.1.103
ProxyChains-3.1 (http://proxychains.sf.net)
|S-chain|-<>-192.168.1.103:1080-<><>-192.168.1.103:22-<><>-OK
pentest@192.168.1.103's password:
Welcome to Ubuntu 14.04.5 LTS (GNU/Linux 3.13.0-161-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

New release '16.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Fri Nov  9 05:29:25 2018 from 192.168.1.103
pentest@ubuntu:~$
```

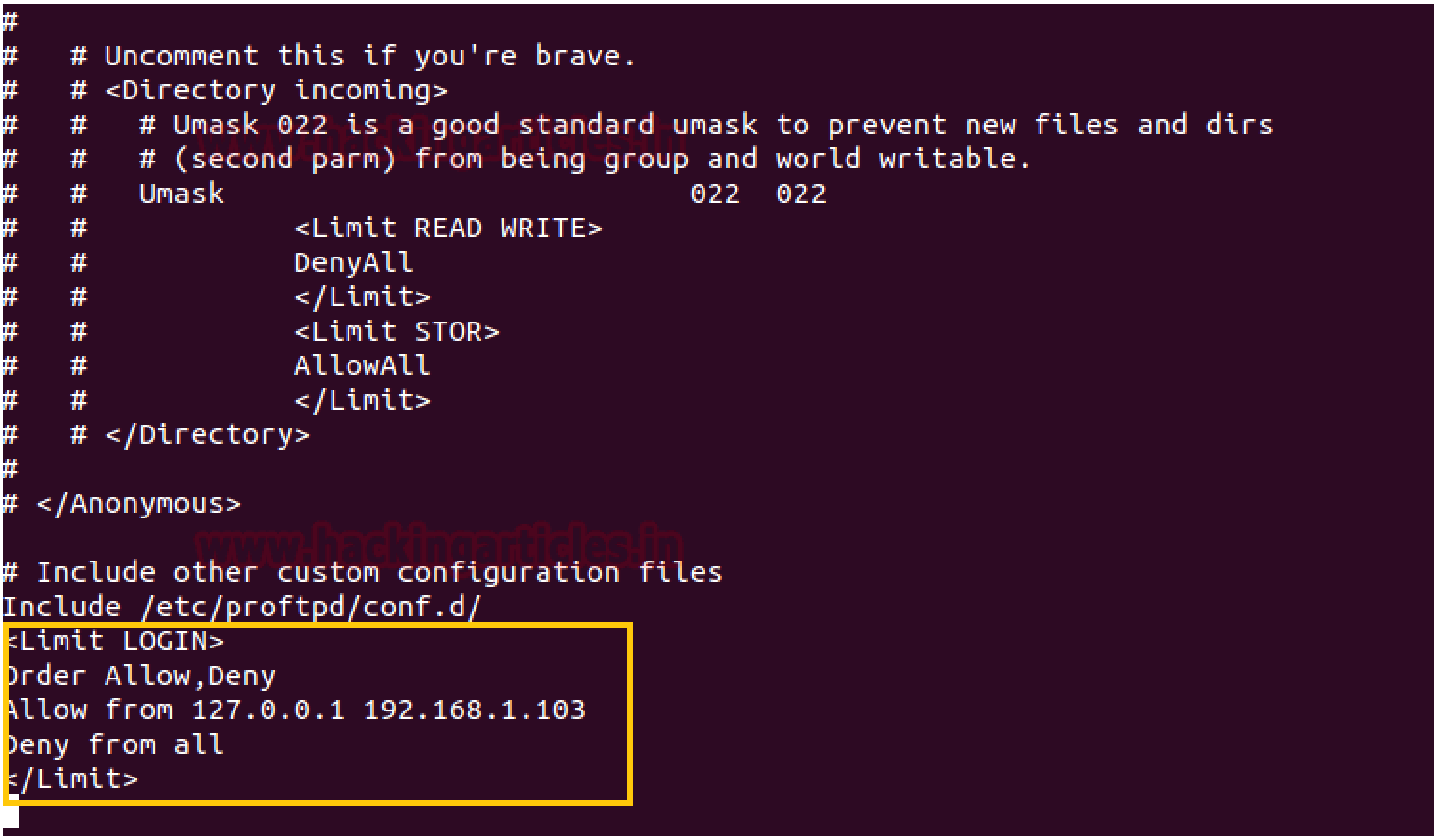
Connecting FTP via Proxy

For connecting FTP via proxy, we have used PRO FTP. SO, you can install it using the following command :

```
apt-get install proftpd
```

Now configuring vsftpd.conf file for FTP Proxy therefore open **/etc/proftpd/proftpd.conf** file and add the following line to allow FTP connection on localhost IP and restrict for other networks.

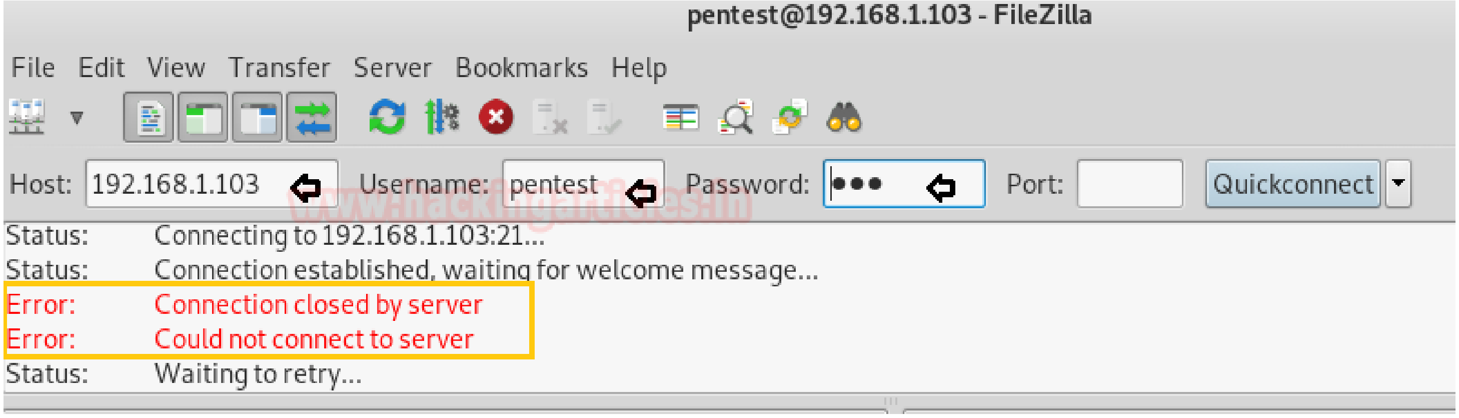
```
<Limit LOGIN>
Order Allow,Deny
Allow from 127.0.0.1 192.168.1.103
Deny from all
</Limit>
```



```
#
#    # Uncomment this if you're brave.
#    # <Directory incoming>
#    #   # Umask 022 is a good standard umask to prevent new files and dirs
#    #   # (second parm) from being group and world writable.
#    #   Umask                           022  022
#    #            <Limit READ WRITE>
#    #            DenyAll
#    #            </Limit>
#    #            <Limit STOR>
#    #            AllowAll
#    #            </Limit>
#    # </Directory>
#
# </Anonymous>

# Include other custom configuration files
Include /etc/proftpd/conf.d/
<Limit LOGIN>
Order Allow,Deny
Allow from 127.0.0.1 192.168.1.103
Deny from all
</Limit>
```

Using FileZilla when we try to connect 192.168.1.103 via port 21 for accessing FTP service, we got an Error "Connection closed by server".

When you face that such type of situation where port 21 is open but you are unable to access it, hence proved the network is running behind a proxy server.

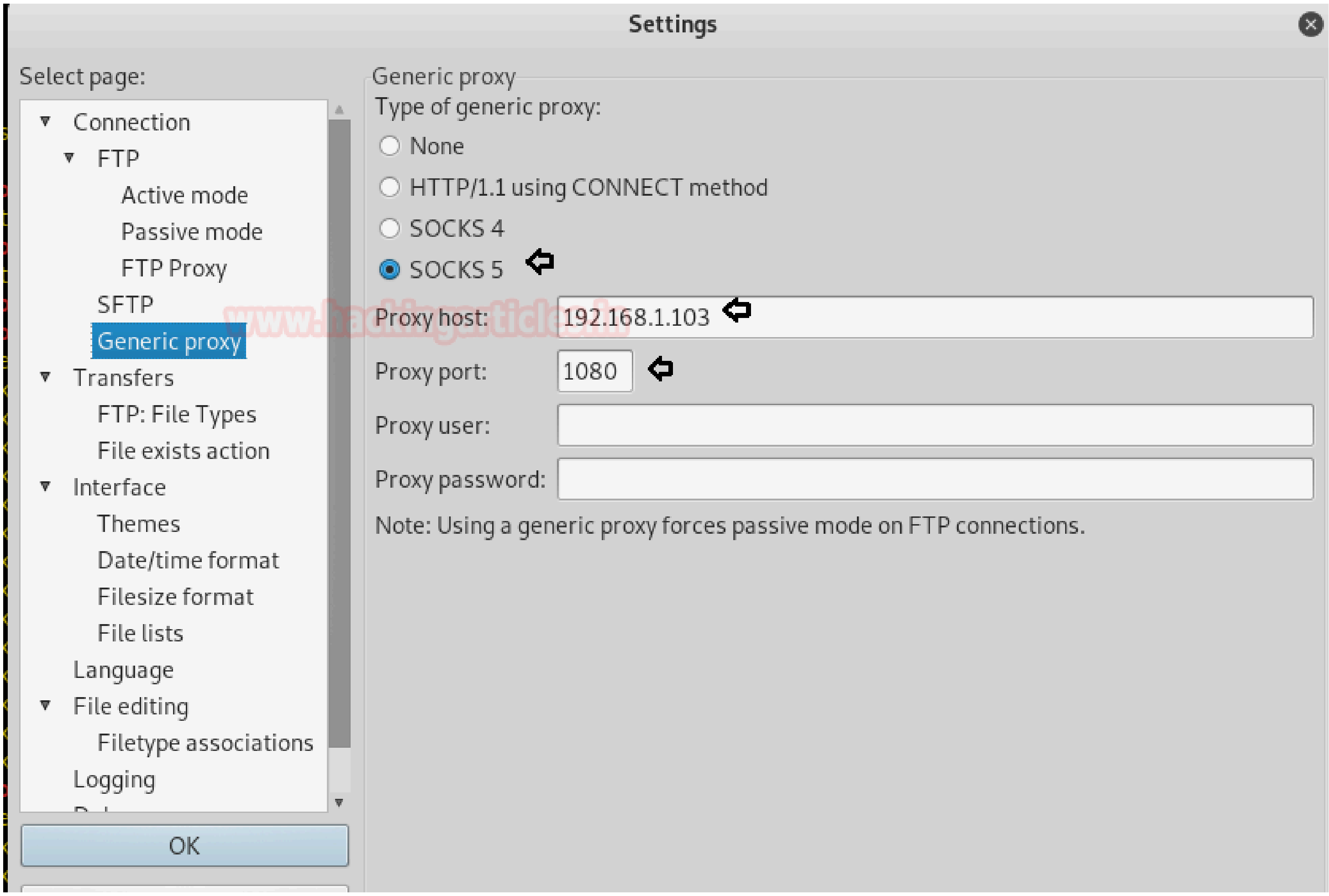


Host: 192.168.1.103 Username: pentest Password: ••• Port:

```
Status:     Connecting to 192.168.1.103:21...
Status:     Connection established, waiting for welcome message...
Error:      Connection closed by server
Error:      Could not connect to server
Status:     Waiting to retry...
```

But FileZilla has multi features as it offers a generic proxy option that forced passive mode on FTP connection. Go to **Settings > Connection > FTP** and select "generic proxy" option and made the following configuration settings.

- Choose SOCKS 5 as generic Proxy
- Proxy HOST IP: 192.168.1.103
- Proxy Port: 1080



Generic proxy

Type of generic proxy:

○ None

○ HTTP/1.1 using CONNECT method

○ SOCKS 4

● SOCKS 5

Proxy host: 192.168.1.103

Proxy port: 1080

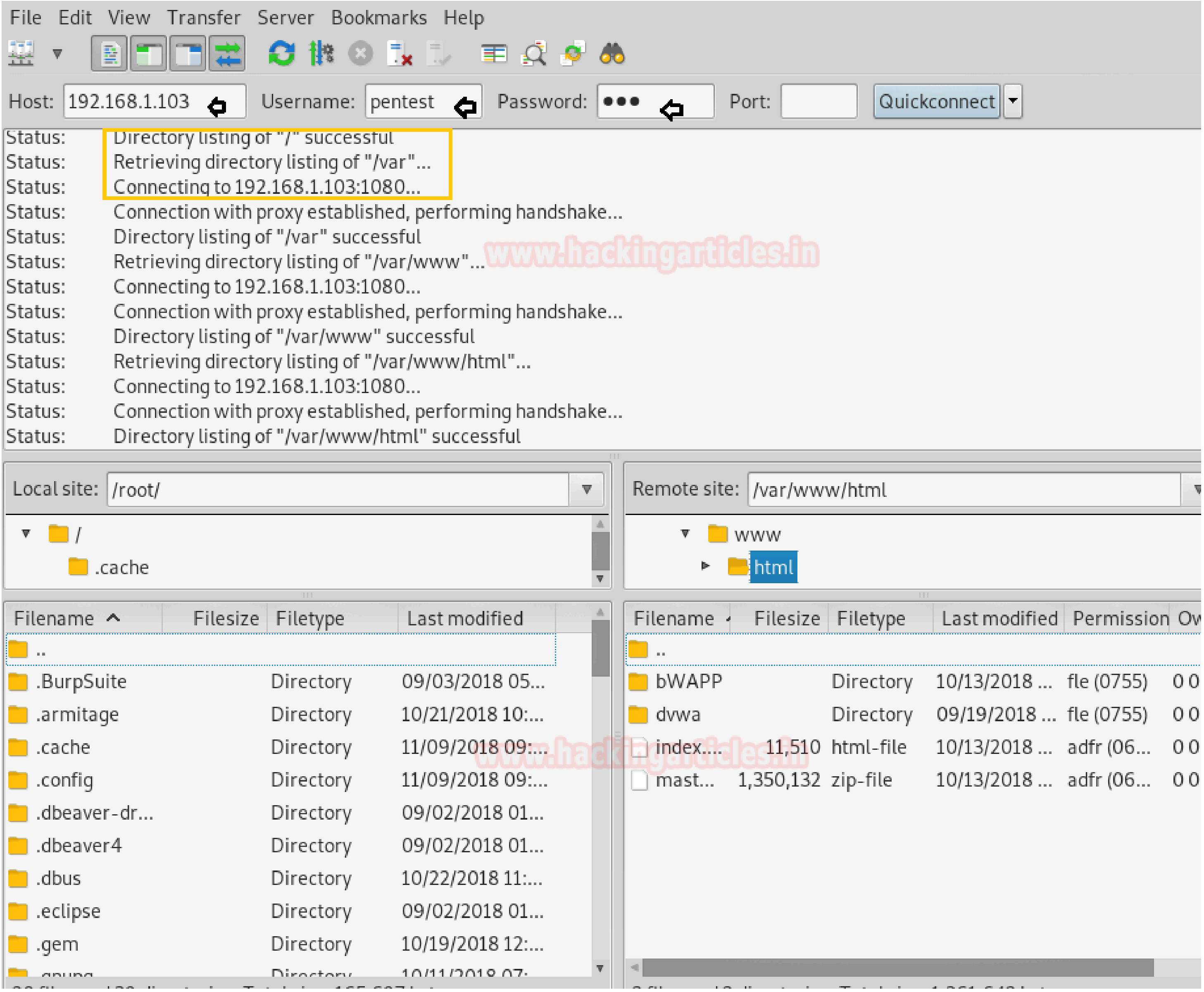
Proxy user:

Proxy password:

Note: Using a generic proxy forces passive mode on FTP connections.

Now again when you will try to connect the target machine via port 21 for accessing FTP service then you will be easily able to access it as shown in the last image.

Hence Proved the SOCKS is actually an all-purpose proxy server and Hopefully, you have found this article very helpful and completely understood the working of Proxy server and another related topic cover in this article.



Host: 192.168.1.103 Username: pentest Password: ●●● Port:

```
Status:     Directory listing of "/" successful
Status:     Retrieving directory listing of "/var"...
Status:     Connecting to 192.168.1.103:1080...
Status:     Connection with proxy established, performing handshake...
Status:     Directory listing of "/var" successful
Status:     Retrieving directory listing of "/var/www"...
Status:     Connecting to 192.168.1.103:1080...
Status:     Connection with proxy established, performing handshake...
Status:     Directory listing of "/var/www" successful
Status:     Retrieving directory listing of "/var/www/html"...
Status:     Connecting to 192.168.1.103:1080...
Status:     Connection with proxy established, performing handshake...
Status:     Directory listing of "/var/www/html" successful
```

Local site: /root/

- /
  - .cache

Filename	Filesize	Filetype	Last modified
..			
.BurpSuite		Directory	09/03/2018 05...
.armitage		Directory	10/21/2018 10:...
.cache		Directory	11/09/2018 09:...
.config		Directory	11/09/2018 09:...
.dbeaver-dr...		Directory	09/02/2018 01...
.dbeaver4		Directory	09/02/2018 01...
.dbus		Directory	10/22/2018 11:...
.eclipse		Directory	09/02/2018 01...
.gem		Directory	10/19/2018 12:...
.gnupg		Directory	10/11/2018 07:...

Remote site: /var/www/html

- www
  - html

Filename	Filesize	Filetype	Last modified	Permission	Ow
..					
bWAPP		Directory	10/13/2018 ...	fle (0755)	0 0
dvwa		Directory	09/19/2018 ...	fle (0755)	0 0
index....	11,510	html-file	10/13/2018 ...	adfr (06...	0 0
mast...	1,350,132	zip-file	10/13/2018 ...	adfr (06...	0 0