THE GEEK STUFF

Linux | DB | Open Source | Web

- Home
- Free eBook
- Contact
- About
- Start Here

C Socket Programming for Linux with a Server and Client Example Code

by Himanshu Arora on December 19, 2011

g+1 24 Like 186 Tweet 31

Typically two processes communicate with each other on a single system through one of the following inter process communication techniques.

- Pipes
- Message queues
- Shared memory

There are several other methods. But the above are some of the very classic ways of interprocess communication.

But have you ever given a thought over how two processes communicate across a network?

For example, when you browse a website, on your local system the process running is your web browser, while on the remote system the process running is the web server. So this is also an inter process communication but the technique through which they communicate with each other is SOCKETS, which is the focus of this article.

What is a SOCKET?

In layman's term, a Socket is an end point of communication between two systems on a network. To be a bit precise, a socket is a combination of IP address and port on one system. So on each system a socket exists for a process interacting with the socket on other system over the network. A combination of local socket and the socket at the remote system is also known a 'Four tuple' or '4-tuple'. Each connection between two processes running at different systems can be uniquely identified through their 4-tuple.

There are two types of network communication models:

1. OSI
2. TCP/IP

While OSI is more of a theoretical model, the TCP/IP networking model is the most popular and widely used.

As explained in our TCP/IP Fundamentals article, the communication over the network in TCP/IP model takes place in form of a client server architecture. ie, the client begins the communication and server follows up and a connection is established.

Sockets can be used in many languages like Java, C++ etc but here in this article, we will understand the socket communication in its purest form (i.e in C programming language)

Lets create a server that continuously runs and sends the date and time as soon as a client connects to it.

Socket Server Example

```
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <string.h>
#include <sys/types.h>
#include <time.h>

int main(int argc, char *argv[])
{
    int listenfd = 0, connfd = 0;
    struct sockaddr_in serv_addr;

    char sendBuff[1025];
    time_t ticks;

    listenfd = socket(AF_INET, SOCK_STREAM, 0);
    memset(&serv_addr, '0', sizeof(serv_addr));
    memset(sendBuff, '0', sizeof(sendBuff));

    serv_addr.sin_family = AF_INET;
    serv_addr.sin_addr.s_addr = htonl(INADDR_ANY);
    serv_addr.sin_port = htons(5000);

    bind(listenfd, (struct sockaddr*)&serv_addr, sizeof(serv_addr));

    listen(listenfd, 10);

    while(1)
    {
        connfd = accept(listenfd, (struct sockaddr*)NULL, NULL);

        ticks = time(NULL);
        snprintf(sendBuff, sizeof(sendBuff), "%.24s\r\n", ctime(&ticks));
        write(connfd, sendBuff, strlen(sendBuff));

        close(connfd);
        sleep(1);
     }
}
```

In the above program, we have created a server. In the code :

- The call to the function ‘socket()’ creates an UN-named socket inside the kernel and returns an integer known as socket descriptor.
- This function takes domain/family as its first argument. For Internet family of IPv4 addresses we use AF_INET.
- The second argument ‘SOCK_STREAM’ specifies that the transport layer protocol that we want should be reliable ie it should have acknowledgement techniques. For example : TCP
- The third argument is generally left zero to let the kernel decide the default protocol to use for this connection. For connection oriented reliable connections, the default protocol used is TCP.
- The call to the function ‘bind()’ assigns the details specified in the structure ‘serv_addr’ to the socket created in the step above. The details include, the family/domain, the interface to listen on(in case the system has multiple interfaces to network) and the port on which the server will wait for the client requests to come.
- The call to the function ‘listen()’ with second argument as ’10′ specifies maximum number of client connections that server will queue for this listening socket.
- After the call to listen(), this socket becomes a fully functional listening socket.
- In the call to accept(), the server is put to sleep and when for an incoming client request, the three way TCP handshake* is complete, the function accept () wakes up and returns the socket descriptor representing the client socket.
- The call to accept() is run in an infinite loop so that the server is always running and the delay or sleep of 1 sec ensures that this server does not eat up all of your CPU processing.
- As soon as server gets a request from client, it prepares the date and time and writes on the client socket through the descriptor returned by accept().

Three way handshake is the procedure that is followed to establish a TCP connection between two remote hosts. We might soon be posting an article on the theoretical aspect of the TCP protocol.

Finally, we compile the code and run the server.

Socket Client Example

```
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <netdb.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <arpa/inet.h>

int main(int argc, char *argv[])
{
    int sockfd = 0, n = 0;
    char recvBuff[1024];
    struct sockaddr_in serv_addr;

    if(argc != 2)
    {
        printf("\n Usage: %s <ip of server> \n",argv[0]);
        return 1;
    }

    memset(recvBuff, '0',sizeof(recvBuff));
    if((sockfd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
    {
        printf("\n Error : Could not create socket \n");
        return 1;
    }

    memset(&serv_addr, '0', sizeof(serv_addr));

    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(5000);

    if(inet_pton(AF_INET, argv[1], &serv_addr.sin_addr)<=0)
    {
        printf("\n inet_pton error occured\n");
        return 1;
    }

    if( connect(sockfd, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0)
    {
       printf("\n Error : Connect Failed \n");
       return 1;
    }

    while ( (n = read(sockfd, recvBuff, sizeof(recvBuff)-1)) > 0)
    {
        recvBuff[n] = 0;
        if(fputs(recvBuff, stdout) == EOF)
        {
            printf("\n Error : Fputs error\n");
        }
    }

    if(n < 0)
    {
        printf("\n Read error \n");
    }

    return 0;
}
```

In the above program, we create a client which will connect to the server and receive date and time from it. In the above piece of code :

- We see that here also, a socket is created through call to socket() function.
- Information like IP address of the remote host and its port is bundled up in a structure and a call to function connect() is made which tries to connect this socket with the socket (IP address and port) of the remote host.
- Note that here we have not bind our client socket on a particular port as client generally use port assigned by kernel as client can have its socket associated with any port but In case of server it has to be a well known socket, so known servers bind to a specific port like HTTP server runs on port 80 etc while there is no such restrictions on clients.
- Once the sockets are connected, the server sends the data (date+time) on clients socket through

clients socket descriptor and client can read it through normal read call on the its socket descriptor.

Now execute the client as shown below.

```
$ ./newsc 127.0.0.1
Sun Dec  18 22:22:14 2011
```

We can see that we successfully got the date and time from server. We need to send the IP address of the server as an argument for this example to run. If you are running both server and client example on the same machine for testing purpose, use the loop back ip address as shown above.

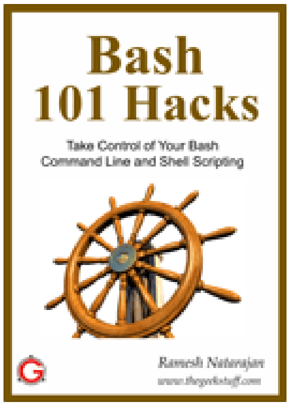
To conclude, In this article we studied the basics of socket programming through a live example that demonstrated communication between a client and server processes capable of running on two different machines.

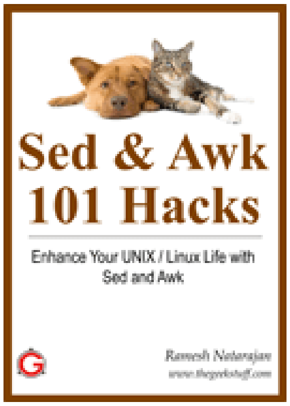
g+1 24 Tweet 31 Like 186 > Add your comment

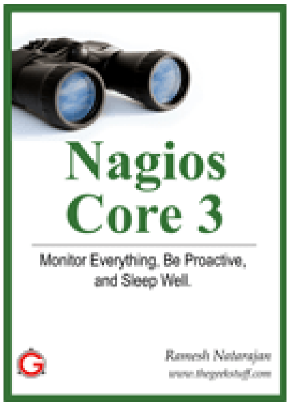
If you enjoyed this article, you might also like..

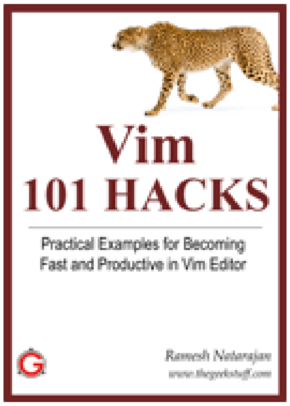
1. 50 Linux Sysadmin Tutorials
2. 50 Most Frequently Used Linux Commands (With Examples)
3. Top 25 Best Linux Performance Monitoring and Debugging Tools
4. Mommy, I found it! – 15 Practical Linux Find Command Examples
5. Linux 101 Hacks 2nd Edition eBook Free

- Awk Introduction – 7 Awk Print Examples
- Advanced Sed Substitution Examples
- 8 Essential Vim Editor Navigation Fundamentals
- 25 Most Frequently Used Linux IPTables Rules Examples
- Turbocharge PuTTY with 12 Powerful Add-Ons








Vim
101 HACKS
Practical Examples for Becoming Fast and Productive in Vim Editor
Ramesh Natarajan
www.thegeekstuff.com

{ 70 comments… read them below or add one }

1 cee December 19, 2011 at 4:18 am

> Thank you for this (yet another) great article. Sometimes I think you can read my mind. Socket programming is exactly what I wanted to take a look at next.
> Keep up the great work with your site/books.
> kind regards
> cee

2 Shanil from Fiji December 19, 2011 at 2:49 pm

> Great Article, good for us engineers here in the Pacific Island countries. Very helpful, Electronics combined with Computing Science. Awesome combination. The Geek Stuff rocks!!

3 Abhijeet Kasurde December 19, 2011 at 10:11 pm

> Great article.Thank you for sharing this information. I would like to hear about new ‘systemd’ in Linux World from GeekStuff

4 Banu December 20, 2011 at 3:02 am

> Awesome article and site. Great information sharing about Unix admin and internals. Looking other great info from Geek!!

5 Himanshu December 21, 2011 at 3:40 am

> Thank you all for your appreciation. 🙂

6 Leslie Satenstein December 22, 2011 at 10:48 pm

> Article is very appropriate. Nice work.
>
> One question. memset(), did you want to initialize the variables with hex 30 (’0′) or did you mean ” or does it make a difference?

7 Himanshu December 23, 2011 at 12:47 am

> @Leslie Satenstein

I think that is a typo. It should have been either ” or 0.
Thanks for pointing out.

8 Leslie Satenstein December 25, 2011 at 1:32 pm

For some reason, the backslash next to the 0 yields a ” on display
I would just put a zero single character into the memset() command.
It gets converted to a hex zero character.

9 Pragmatic Programmer December 28, 2011 at 11:50 pm

Great stuff Himanshu! This is a very good primer for Socket programming.
One suggestion here (for the coding style, nothing related to Socket programming)

Instead of
char sendBuff[1025];
and then
memset(sendBuff, ’0′, sizeof(sendBuff));

char sendBuff[1025] = {0};
works equally well.
I know this is not a coding tutorial, but just mentioned hope you don’t mind 🙂

Cheers.

10 Himanshu December 29, 2011 at 1:28 am

@Pragmatic Programmer

Hey thanks for the optimization. Yes, its a better way to achieve the desired value in array. Thanks for your input.

Cheers!!!

11 deepak kumar January 2, 2012 at 6:33 am

nice article.helpful

12 himanshu January 2, 2012 at 9:48 am

@deepak

Thanks!!!!

13 Chetan March 20, 2012 at 12:31 pm

i liked it, and thank you for helping me through this.

14 Vijay Kanta April 9, 2012 at 4:11 am

This was what I was looking for, and it served my purpose almost. Good one!

15 prashant dubey (varanasi) June 21, 2012 at 10:55 pm

this is to good program for the learning student because this program syntax is user friendly…….
thanks to upload…..
(” MCA “) with IGNOU

16 kavya July 23, 2012 at 10:55 am

hey frnds..! i wanna learn “C socket programming” am a beginner so suggest some websites for me..!
B.Tech IT

17 Himanshu July 23, 2012 at 11:02 pm

@kavya
You can start here :
http://www.linuxjournal.com/article/2333

18 agnel brilya August 18, 2012 at 4:01 am

good nd thanx a lot 🙂

19 nicks August 23, 2012 at 2:41 am

Helping a lot during Boring Lab

20 Pawan September 4, 2012 at 7:18 am

Hi, Himanshu,
I am totally new to Socket programming. I have few questions, Is the both Server and Client code .c files are on same machine? If Yes, Did you compile both .c files together? What exactly is this : ./newsc

Thanks

21 Farhad September 12, 2012 at 1:21 am

Hi Himanshu,
i just tried to make few changes in the server program to reply back to the client with “pinged” data instead of “Date+Time” .

This is the changes i have made in server program,but one problem is,i should get the pined data on client console(like Date+Time),but i am getting the pinged data on the server console only.

```
#include // change made
int main(int argc, char *argv[])
{
int listenfd = 0, connfd = 0;
struct sockaddr_in serv_addr;

char sendBuff[1025];
// time_t ticks; //change made

listenfd = socket(AF_INET, SOCK_STREAM, 0);
memset(&serv_addr, ’0′, sizeof(serv_addr));
memset(sendBuff, ’0′, sizeof(sendBuff));

serv_addr.sin_family = AF_INET;
serv_addr.sin_addr.s_addr = htonl(INADDR_ANY);
serv_addr.sin_port = htons(5000);

bind(listenfd, (struct sockaddr*)&serv_addr, sizeof(serv_addr));

listen(listenfd, 10);

while(1)
{
connfd = accept(listenfd, (struct sockaddr*)NULL, NULL);

// ticks = time(NULL); //change made
// snprintf(sendBuff, sizeof(sendBuff), “%.24s\r\n”, ctime(&ticks)); //change made
system(“ping NULL”); //change made
snprintf(sendBuff, sizeof(sendBuff), “%.24d\r\n”, system(“ping 10.0.2.15″)); //change made

write(connfd, sendBuff, strlen(sendBuff));

close(connfd);
sleep(1);
}
}
```

22 Himanshu Arora September 12, 2012 at 10:14 am

@Farhad

Well, I think that the following line is wrong :

snprintf(sendBuff, sizeof(sendBuff), “%.24d\r\n”, system(“ping 10.0.2.15″)); //change made

I think you should first go through the man pages of snprintf() function and system() function to understand how are they used.

23 jayasri September 20, 2012 at 12:04 am

Thanks,Good article(very useful for freshers).

24 bbb October 2, 2012 at 7:59 am

it was help full …. try to explain inet_pton() function and its arguments in detail

25 rosy October 19, 2012 at 2:53 am

thnx alot it’s Helping a lot during Boring lecture , but i need som help in assignment
that need ;
The client sends the server a 16-bit integer (short) that represents the number of integers that it will follow in the input. For example, if the client sends the server 9, then that means that the client is

going to send the server 9 integer numbers. Then, the server computes the sum and average of the

if you can help me .. 🙂 thnx alot
numbers sent by the client and sends them (i.e., the sum and average) to the client. Finally, the client
should receive those results and print them out
Your input should be read from the keyboard as an ASCII character string.

26 Farah October 20, 2012 at 3:53 am

Very nice tutorial Himanshu 🙂

Unfortunately it’s not working with me. Maybe you can help me out a bit.
I installed ubuntu on a virtual machine. I have windows installed on my laptop. I wanted to see if I can get the date and time from my windows and display them on ubuntu.
I compiled the code and it gave no errors, but when I wrote ./Socket-1 it gave me this error:
Connection Failed.

Could it be because I am running this program on a linux virtual machine?

Thanks 🙂

27 ajesh October 28, 2012 at 10:46 am

Thank you, it is a great article.. 🙂

28 Farhad October 31, 2012 at 10:14 pm

Hi guys…can we create a loop in our socket program where the client can continously query for data n server responds to each query,without closing the sockets?
where should the loop be?

and my second question is ,how can i send a trap signal from server to client in socket programming?

29 biswa parida November 1, 2012 at 1:11 am

i appreciate the comments

30 Monstre November 10, 2012 at 11:48 pm

You are a genius 😀 !!! I appreciate a lot this simply code working!!!

31 Angela November 12, 2012 at 1:15 pm

Thank you for the great article. I have a question: how can we let the server run forever and continuously send/receive data to/from the client? I.e., is there anyway to not close the socket? Something similar to serve_forever() in Python?

32 Jurij November 20, 2012 at 3:12 pm

Great tutorial. I have a question. What exactly means variables “sockfd”, “connfd”, “listenfd” and for what we use they. Thanks

33 Heena December 24, 2012 at 12:54 am

Nice tutorials !!! really helpful in learning socket prog….thanx

34 Supriya Magdum January 9, 2013 at 12:32 am

AWESOME SITE……

35 charu January 10, 2013 at 12:29 am

hey gr8 work ya….dats very helpful…thanku…

36 Prashant January 10, 2013 at 10:14 am

Hey can u give me exact sequence to run this program

37 Vaibhav February 9, 2013 at 3:51 pm

Best Tutorial on web to learn socket programming.

Thank you !

38 Farhad February 11, 2013 at 12:26 am

Situation:my client and server both sockets are open and connected.
Can i send an alarm/trap from server to client for a specific task without the client querying for anything?

Your help will be highly appreciated.!

39 Suneel Kumar February 22, 2013 at 12:27 pm

Thankyou very much .. I want to ask about "server performs some computational task on behalf of Clients and response to him " like as example client send two integers to server and server add those number and output response to server.. How I can send Integers, command string.. etc in which way on server as well as client.
Please reply

40 sreelakshmi March 25, 2013 at 3:02 am

sockt.c: line 12: syntax error near unexpected token `('
sockt.c: line 12: `int main(int argc, char *argv[])'
there is an error like what can i do?

41 Yash Rohilla April 1, 2013 at 9:59 am

Great article Himanshu! The explanations beneath the programs really helped me understand the concept of socket progamming. I'm still getting an issue though. I've been trying to run the server and client on the same machine, so I've been using the loopback address. However, I keep getting a connect failed output. Any ideas?

42 Farhad April 3, 2013 at 4:04 am

Hi friends..
I did a Server/client model using socket programming.

I am having some trouble in my server side program
I am having an infinte while loop on server side and I am checking few conditions on server side. If the conditions satisfy then the while loop goes on doing its work in each iteration ,however if any condition fails then the client side program ends but at server side it shows an infinte loop running without any exit.

I am trying to give
close(accept_sd);
close(listen_sd);
break;
commands for false conditions too,but the loop keeps on going infinitely .

Kindly help..!
Thank You

43 RICHA April 16, 2013 at 12:35 am

how to implement quiz between client and server using socket?

44 umair August 7, 2013 at 10:15 am

plz help me how can i send Hi(sent) to server side and Hi(received) to client side using this code?

45 Rajesh September 10, 2013 at 10:38 am

Thank you !!!

Nice Article…

please change this : memset(recvBuff, '0′,sizeof(recvBuff));
with : memset(recvBuff, '',sizeof(recvBuff)); or memset(recvBuff, 0,sizeof(recvBuff));

46 lsatenstein September 15, 2013 at 6:55 am

Definitely not the memset with "" as 2nd argument. with "" as 2nd argument, you would be telling memset to use the address of ""

Perhaps the 2nd argument should be. *"". (The contents of the null string)

47 kaushik September 16, 2013 at 4:39 am

Kindly use comments for better understandablity……..ammendments will be appreciated
thank u

48 Nikhil September 27, 2013 at 2:19 am

Hi I tried running ur code with the following changes in the while loop

```
while(1)
{
connfd = accept(listenfd, (struct sockaddr*)NULL, NULL);

// ticks = time(NULL);
snprintf(sendBuff, sizeof(sendBuff), “*****************so m here !!@!@”);
write(connfd, sendBuff, strlen(sendBuff));
```

For the first time this code runs fine .. and then I shutdown the server..
But now when i try to run the client with the server ip it still gives me the same output ..

can u guess where is it going wrong ??

49 Diwakar October 9, 2013 at 1:13 pm

Hi,
Can you provide any examples of socket programming codes from real world i.e. industry? Like how its used in industry preferably in hardware interaction paradigm e.g in relation to network equipments like DSLAM/ONT?

50 archana October 16, 2013 at 7:01 am

Hi,
I am doing c programming with unix…I have copied ur code and trying to run on same machine….then I am getting error like “Connect Failed”..so how can we resolve this?

51 Joris Achten January 3, 2014 at 2:54 am

Nice one!

52 Leslie Satenstein January 4, 2014 at 8:42 pm

In the above examples
memset(recvBuff, ’0′,sizeof(recvBuff));
puts a load of zeros into the string.

The following corrections may be made as

memset(recvBuff, 0,sizeof(recvBuff));
or
memset(recvBuff, (char) 0,sizeof(recvBuff));

or
memset(recvBuff, ”,sizeof(recvBuff)); // a \ followed by a 0

53 Leslie Satenstein January 4, 2014 at 8:45 pm

the sequence ‘ followed by \ followed by 0 folloed by ? yields the same as

does not display properly so ue the following

(char) 0

54 bonzz January 21, 2014 at 12:12 am

What if we have to give some message instead of time..

55 aniket January 29, 2014 at 5:01 am

nice ,
it helps me in my colg work
thnks,….

56 BhanuPriya.s February 17, 2014 at 2:49 am

thanks.I really appreciate …
thankyou..

57 Tamilselvi February 17, 2014 at 2:51 am

very nice coding.thank u…….

58 Said March 30, 2014 at 9:43 am

Plz Can U send Me The AF_UNIX ,SOCK-DGRAM Code 🙂

59 nikita April 9, 2014 at 2:59 pm

can please send me the code using send and recv calls instead of read and write?
i have run both the codes for server as well as for client….bt the server code never gets quit even if i enter the choice for quit….so suggest me in this
thank you..!!!!

60 Leslie Satenstein April 10, 2014 at 8:31 am

Hi Nikita

Please type man socket or info socket. Google socket programming.
When you quit a program you should of course do first do a close() call for the socket that was opened.

61 Douglas April 18, 2014 at 8:41 pm

You are not checking for errors, I lost almost an hour because of an error on bind(). Permission denied. Please correct the code.

62 giordano April 23, 2014 at 9:44 am

hi,
I am doing a patch to a very old program running on sco unix 5.0.5 server.
i need a server that listen on tcp port from other hosts but after many tests
the wrong point is on accept call: the accept dont wait client connection but return with a 0 value and errno is set to "unknown error".
sockets are blocking as default o no ? … or i must set flag to blocking beaviour?

63 Sam April 25, 2014 at 7:56 am

You say, “In the call to accept(), the server is put to sleep and when for an incoming client request, …” Instead of “and when” did you mean “waiting”?

64 fra May 16, 2014 at 1:24 pm

Hey.

Thanks for the code snippet! Very helpful.
Questions: Can I connect over any port that I like? I have some data acquisition equipment that puts out data at a given IP over port 55555. I have tried compiling and running with different ports and I connection failures.

Thanks!

65 n.a.s May 18, 2014 at 6:04 pm

Hi,
How I can pass array to server over UDP?

Thanks

66 Azam May 29, 2014 at 8:36 pm

Hi,I wanna show CPU usage,memory usage,filesystem read & write in sockt programing with C in Ubuntu.
please help me
thanks in advance

67 iliya June 15, 2014 at 3:15 pm

hi,
good one . The example is very simple and also understandable but there is a problem same as “archana” mentioned . i’ve got the problem exactly the same . how can i fix the problem ? (Error : Connect Failed)

Thanks alot.

68 Andy July 9, 2014 at 12:08 am

This a late comment, but you deserve big kudos for this program set and explanation. This is of great benefit to my offensive security training. I need to understand the inner workings of exploits and payloads, which contain socket-related activity. This is perfect for my current studies.

69 padmanagarajan July 9, 2014 at 12:24 am

Quite easy to understand.. thanks a lot…

70 Pragya Daga September 2, 2014 at 12:01 pm

I’m new to using linux and socket
can you please tell me what does /newsc 127.0.0.1 do?
it shows no directory present

Leave a Comment

Name

E-mail

Website

□ Notify me of followup comments via e-mail

Submit

Previous post: Perl and Excel: How to Read, Write, Parse Excel Files using Perl

Next post: How to Check and Repair MySQL Tables Using Mysqlcheck

- RSS | Email | Twitter | Facebook | Google+

  Search

-

- **COURSE**

  - Linux Sysadmin CentOS 6 Course - Master the Tools, Configure it Right, and be Lazy

- **EBOOKS**

  - Free Linux 101 Hacks 2nd Edition eBook - Practical Examples to Build a Strong Foundation in Linux
  - Bash 101 Hacks eBook - Take Control of Your Bash Command Line and Shell Scripting
  - Sed and Awk 101 Hacks eBook - Enhance Your UNIX / Linux Life with Sed and Awk
  - Vim 101 Hacks eBook - Practical Examples for Becoming Fast and Productive in Vim Editor
  - Nagios Core 3 eBook - Monitor Everything, Be Proactive, and Sleep Well

-

The Geek Stuff

Like

8,165 people like The Geek Stuff.

- **POPULAR POSTS**

  - 12 Amazing and Essential Linux Books To Enrich Your Brain and Library
  - 50 UNIX / Linux Sysadmin Tutorials
  - 50 Most Frequently Used UNIX / Linux Commands (With Examples)
  - How To Be Productive and Get Things Done Using GTD
  - 30 Things To Do When you are Bored and have a Computer
  - Linux Directory Structure (File System Structure) Explained with Examples
  - Linux Crontab: 15 Awesome Cron Job Examples
  - Get a Grip on the Grep! – 15 Practical Grep Command Examples
  - Unix LS Command: 15 Practical Examples
  - 15 Examples To Master Linux Command Line History
  - Top 10 Open Source Bug Tracking System
  - Vi and Vim Macro Tutorial: How To Record and Play
  - Mommy, I found it! -- 15 Practical Linux Find Command Examples
  - 15 Awesome Gmail Tips and Tricks
  - 15 Awesome Google Search Tips and Tricks
  - RAID 0, RAID 1, RAID 5, RAID 10 Explained with Diagrams
  - Can You Top This? 15 Practical Linux Top Command Examples
  - Top 5 Best System Monitoring Tools
  - Top 5 Best Linux OS Distributions
  - How To Monitor Remote Linux Host using Nagios 3.0
  - Awk Introduction Tutorial – 7 Awk Print Examples
  - How to Backup Linux? 15 rsync Command Examples
  - The Ultimate Wget Download Guide With 15 Awesome Examples
  - Top 5 Best Linux Text Editors
  - Packet Analyzer: 15 TCPDUMP Command Examples
  - The Ultimate Bash Array Tutorial with 15 Examples
  - 3 Steps to Perform SSH Login Without Password Using ssh-keygen & ssh-copy-id
  - Unix Sed Tutorial: Advanced Sed Substitution Examples
  - UNIX / Linux: 10 Netstat Command Examples
  - The Ultimate Guide for Creating Strong Passwords
  - 6 Steps to Secure Your Home Wireless Network
  - Turbocharge PuTTY with 12 Powerful Add-Ons

- ## CATEGORIES

  - Linux Tutorials
  - Vim Editor
  - Sed Scripting
  - Awk Scripting
  - Bash Shell Scripting
  - Nagios Monitoring
  - OpenSSH
  - IPTables Firewall
  - Apache Web Server
  - MySQL Database
  - Perl Programming
  - Google Tutorials
  - Ubuntu Tutorials
  - PostgreSQL DB
  - Hello World Examples
  - C Programming
  - C++ Programming
  - DELL Server Tutorials
  - Oracle Database
  - VMware Tutorials

- Ramesh Natarajan

  g+ Follow

- ## About The Geek Stuff

My name is **Ramesh Natarajan**. I will be posting instruction guides, how-to, troubleshooting tips and tricks on Linux, database, hardware, security and web. My focus is to write articles that will either teach you or help you resolve a problem. Read more about Ramesh Natarajan and the blog.

- ## Support Us

Support this blog by purchasing one of my ebooks.

Bash 101 Hacks eBook

Sed and Awk 101 Hacks eBook

Vim 101 Hacks eBook

Nagios Core 3 eBook

- **Contact Us**

Email Me : Use this Contact Form to get in touch me with your comments, questions or suggestions about this site. You can also simply drop me a line to say hello!.

Follow us on Google+

Follow us on Twitter

Become a fan on Facebook