How are passwords stored in Linux (Understanding hashing with shadow utils)

Submitted by Sarath Pillai on Wed, 04/24/2013 - 16:57

A user account with a corresponding password for that account, is the primary mechanism that can be used for getting access to a Linux machine. Its very much logical to think that the passwords of all the user's in a system must first be saved in some kind of a file or a database, so that it can be verified during a user login attempt.

And you do not require the skill set and expertise of a computer security scientist to think rationally that if you get hold of that database or file, which stores all the passwords, you can easily get access to the machine.

Some people will argue at this point that the database, or the file that contains the passwords, are in an encoded format (hash value), and its not possible to know the real password, from the encoded password hash. However its possible to detect the real password from the encoded ones, by doing a dictionary attack against the encoded value. Let's get back to some basics before going ahead.

Hashing is a mathematical method to produce a fixed length encoded string for any given string. The main strength of the hashing algorithm is the fact that, you cannot detect the original string from the encoded string. In other words you can call it one way hashing. hashing algorithm's are not only used for storing passwords but also used for data integrity check. You will get a unique fixed length encoded string for any data you give, and that encoded string will be unique to that particular data.

In short no two data can have the same hash (encoded string). Changing a single character of the data will produce a completely different encoded hash(Because the data is different). The strength of hashing is used in almost all communication protocol's in some or the other way to check and confirm the originality and integrity of the data(because, if you alter the data the hash will change).

But what if an attacker gets hold of the encrypted hash and goes on trying different combination of words to produce the exact same hash. With the advanced computing available these days, the attacker can go through millions of combinations in hours. This risk was also applicable to the way passwords were stored in UNIX/Linux system's. Although the passwords were encoded, if an attacker get's hold of that password file, then he can attempt to break the password.

From the initial days of Unix, passwords were stored in a file called as **/etc/password**. And a major loophole in this single file, is that the file is world readable. Which means any user in the system can read that file. This was done purposely because the file contains critical user related information other than passwords, and many applications and system tools require that information to function properly. Lets see the permission of the /etc/password file in linux.

```
[root@slashroot1 ~]# ll /etc/passwd
-rw-r--r-- 1 root root 1875 Dec 14 23:17 /etc/passwd
[root@slashroot1 ~]#
```

You can see that there is an **"r"**(Which stands for read), in all the three fields for that file. Let's try to change this permission, so that only **"root"** can read the file.

```
[root@slashroot1 ~]# chmod 600 /etc/passwd
[root@slashroot1 ~]# ll /etc/passwd
-rw------- 1 root root 1875 Dec 14 23:17 /etc/passwd
[root@slashroot1 ~]#
```

Now let's see what's the effect of this change on the system...Let's try to become a normal user, and see what happens.

```
[root@slashroot1 ~]# su - sarath
su: warning: cannot change directory to /home/sarath: Permission denied
id: cannot find name for user ID 500
-bash: /home/sarath/.bash_profile: Permission denied
-bash-3.2$
```

oh did you see that...You cant even change the user. Even if you change the user, you will not get the user name, home directory, custom shell etc (Because all these details are stored in **/etc/passwd**). Due to this reason the file **/etc/passwd,** needs to be kept world readable. But we cannot keep passwords in a file that's world readable(because of the risk involved, even though its encoded in a one way hash algorithm)

Hence there arises a need to separate passwords from this file and keep it in a file, that's only accessible by root. The solution to this problem is implemented in the form of a package in Linux called **"shadow-utils".**

What is shadow-utils package in Linux?

Shadow utils is a package in Linux that's installed by default in most of the distributions, used for separating passwords from **/etc/passwd**. After implementing shadow-utils, passwords are now saved in **/etc/shadow** file in Linux. This **/etc/shadow** file is only accessible by root. Let's see the contents of the /etc/shadow file, and also its permission.

```
[root@slashroot1 ~]# ll /etc/shadow
-r-------- 1 root root 1140 Dec 14 23:17 /etc/shadow
[root@slashroot1 ~]#
```

You can see that unlike the **/etc/passwd** file the **/etc/shadow** file only has the **"r" (read)** permission set for root user. Which means no other user has access to this file. Let's see what's the content of this file.

```
[root@slashroot1 ~]# cat /etc/shadow
root:$1$Etg2ExUZ$F9NTP7omafhKIlqaBMqng1:15651:0:99999:7:::
```

Let's understand each and every field of that output, that are separated by a **":".**

1. The first field is self explanatory, its the USERNAME

2. The second field is the encoded password (Which is a one way hash..we will be discussing this in detail)

3. The third field is the day's since the UNIX time that password was changed.

Refer: What is UNIX time? (http://en.wikipedia.org/wiki/Unix_time)

4. This field specifies the number of days, that are required between password changes.

5.No of days after which its necessary to change the password.

6.This is the number of days before the required password change, the user gets a warning

7.If the password has expired, after this number of days the account will be disabled

8.No of days from the Unix Time, the account is disabled

9. This field is not used yet...

Now you will be confused, that why does the **/etc/shadow,** file contains these many information's rather than only the encoded password. This is because **shadow-util's** package provides some more advanced feature's along with storing encoded passwords in **/etc/shadow**. The above mentioned fields of **/etc/shadow,** file tell's those added feature's to a certain extent like **age of the passwords and its expiry**, and also below mentioned feature's.

- Default parametres for user account creation (**/etc/login.defs**)
- Tools to modify user accounts and groups
- Enforcing strict password selection

How is an encoded password stored in /etc/shadow file, and how does the system verify a user typed password during login?

For understanding how this entire thing works, let's take the case of the previously shown example entry for root user, from **/etc/shadow** file.

```
[root@slashroot1 ~]# cat /etc/shadow
root:$1$Etg2ExUZ$F9NTP7omafhKIlqaBMqng1:15651:0:99999:7:::
```

From the above shown example entry, our topic of interest is the second field(the field with the encoded hash of the password).

1Etg2ExUZ$F9NTP7omafhKIlqaBMqng1

The above shown encoded hash value can be further classified into three different fields as below.

1. The first field is a numerical number that tell's you the hashing algorithm that's being used.

- $1 = MD5 hashing algorithm.
- $2 =Blowfish Algorithm is in use.
- $2a=eksblowfish Algorithm
- $5 =SHA-256 Algorithm
- $6 =SHA-512 Algorithm

2. The second field is the salt value

Salt value is nothing but a random data that's generated to combine with the original password, inorder to

increase the strength of the hash..

3.The last field is the hash value of salt+user password (we will be discussing this shortly).

So in our example entry of root, as shown below,

1Etg2ExUZ$F9NTP7omafhKIlqaBMqng1

The above shown encoded password is using MD5 hashing algorithm (because the of **1**)

Salt value is **Etg2ExUZ (the content between the second and third $ sign)**

And the hash value of "**PASSWORD + SALT"**. Let' s reproduce the same output by providing the salt value of **Etg2ExUZ** and the original password.

```
[root@slashroot1 ~]# openssl passwd -1 -salt Etg2ExUZ redhat
$1$Etg2ExUZ$F9NTP7omafhKIlqaBMqng1
[root@slashroot1 ~]#
```

With the help of the above openssl command, you can see that the encoded entry can only be reproduced with the exact same salt value (which is always randomly selected by the password program).

This is what is done by the login program when you enter the password, it uses the salt value and your entered password to create an encoded string. If that encoded string matches the encoded string from the shadow file, then the user login is considered as successful.

Changing the salt will change the entry in shadow file. **-1** option used in the above command, tell's which hashing algorithm to use(1 indicates md5 algorithm).

What will happen if there is no salt value at all?

- Salt value is a major component that strengthens the way a linux system stores password. Imagine that there is no salt value applied before storing passwords in linux. As we have discussed in the beginning of this article, a dictionary attack with common dictionary words will become much more easier to do.

By using the salt value(which is randomly generated while generating passwords), an attacker needs to go throgh different combinations of salt values as well as password string's to guess what the original password is.

- An attacker cannot easily guess that two user's are using same passwords. Because even if the attacker has somehow gained access to the shadow file, he cannot say looking at two encoded passwords, that they are using the same password. This is because both of them will be having different salt values.

How to Display hashing Algorthm used in your Linux Machine?

```
[root@localhost ~]# authconfig --test|grep hashing
 password hashing algorithm is md5
```

Above command clearly shows that, at present the algorithm used by your Linux machine is md5(Which will be used for all the user's by default).

How will you configure your Linux machine to use SHA-512 hashing algorithm?

```
[root@localhost ~]# authconfig --passalgo=sha512 --update
```

Please note the fact that unless you change the password of your users, SHA-512 hashing algorithm will not be updated for already existing users.

You can also force your Linux users to change their passwords during the next login by using chage command(which is also provided by the shadow-utils package).

I am doing it for the user tiwary, so that user will need to compulsorily, change the password on next login.

```
[root@localhost ~]# chage -d 0 tiwary
```

Above command will force user tiwary to change his password at next Login.

Now when you see /etc/shadow file you can see that the user tiwary, has a different algorithm than all other user's after the password change.

```
[root@localhost ~]# cat /etc/shadow
root:$1$flVALfyK$kJfaoYnsAm7/plT3.PCmJ/:15816:0:99999:7:::
satish::15804:0:99999:7:::
slashroot:$1$MIyV9col$Up9YON8Z.TI1x37xgFvuO0:15804:0:99999:7:::
u1::15813:0:99999:7:::
himanshu:$1$0Iwvz7CA$QOJLfOSJZuSLC19LSFxt1.:15810:0:99999:7

tiwary:<span style="color:#ff0000;"><strong>$6$</strong></span>QXBJkK8N$owDISwfo1wMH1BqoL9Rx/RD4
```

In the above shown output 6 stands for sha512 algorithm.

How to generate a shadow style password hash?

```
[root@localhost ~]# openssl passwd -1 redhat123
$1$jp5rCMS4$mhvf4utonDubW5M00z0Ow0
```

In this case salt value is the eight characters between the 2nd and 3rd $ sign. i.e `jp5rCMS4.`
You can paste the above output into your shadow file for a particular user and then that user can easily login with the password "redhat123".

Rate this article:
Average: 4.1 (727 votes)

1 Comment Sort by Oldest

Add a comment...

கமலக் கண்ணன்

053bd48f088c928b8408908be59d76c5e37d974e163ef92c8a3e25f3dd0b713e7851fe6d103cd270028d527ef0763644a4629c807ad37eb1446f8048a971f360

how to decrypt hash sir this hash encrypt please decrypt method kali linux

Like · Reply · 35w

Facebook Comments Plugin

Add new comment (/how-are-passwords-stored-linux-understanding-hashing-shadow-utils#comment-form)

Comments

That was very informative.. (/comment/76#comment-76)

Permalink (/comment/76#comment-76) Submitted by Siddharth on Thu, 06/06/2013 - 15:30

That was very informative..

In this section where we can force Linux users to change their passwords during the next login by using chage command, i got an unusual situation where my linux(RHEL6) allow me to change password at login time but it is not accepting new password.
Every time i set new password an error message will prompt saying "Bad password" or "Too short" and after some try it will eventually be exhausted.

help me through this..

reply (/comment/reply/173/76)

Hi siddharth,Good to know (/comment/77#comment-77)

Permalink (/comment/77#comment-77) Submitted by Sarath Pillai on Fri, 06/07/2013 - 17:06

Hi siddharth,

Good to know that the article was informative...So let's get back to the problem faced by you in Red Hat Enterprise Linux 6 version, for password change.

According to the error codes you mentioned ("Bad Password" & "Too short"), it appears to me

that RHEL 6 has by default, a strict password complexity enabled through PAM. PAM stands for Pluggable Authentication Module, and is responsible for different authentication schemes under Linux. The basic file that helps you to configure password complexity, or i must say the file that has the module and options, which enables password complexity is **/etc/pam.d/system-auth**.

Let's have a look at the contents of that file(Please note the fact that am using RHEL 5 instead of 6, for showing you this example. Most red hat versions have the same contents inside this file.)

```
[root@myvm1 ~]# cat /etc/pam.d/system-auth
#%PAM-1.0
# This file is auto-generated.
# User changes will be destroyed the next time authconfig is run.
auth        required      pam_env.so
auth        sufficient    pam_unix.so nullok try_first_pass
auth        requisite     pam_succeed_if.so uid >= 500 quiet
auth        required      pam_deny.so

account     required      pam_unix.so
account     sufficient    pam_succeed_if.so uid < 500 quiet
account     required      pam_permit.so

password    requisite     pam_cracklib.so try_first_pass retry=3
password    sufficient    pam_unix.so md5 shadow nullok try_first_pass
password    required      pam_deny.so

session     optional      pam_keyinit.so revoke
session     required      pam_limits.so
session     [success=1 default=ignore] pam_succeed_if.so service in cr
session     required      pam_unix.so
```

The line that matters to us as far as password complexity is considered is

```
password    requisite     pam_cracklib.so try_first_pass retry=3
```

Red Hat Enterprise Linux 6(or the one in which you are getting those errors while setting up password), must have a couple of other parameters as well, due to which you are getting a strict password policy enforced. You might see something like the below as arguments to the above shown line.

minlength, lcredit, ucredit, dcredit, ocredit. Each of these mentioned arguments has got their own meanings, that makes up the required complexity in the password.

minlength=10 specifies a minimum password length of 10 letters, **lcredit=3** specifies that the password must have 3 lover case letters, **ucredit=3** specifies that the password must have 3 upper case letters, **dcredit=3** specifies there must be atleast 3 digits in the password, **ocredit=3** specifies there must be atleast 3 other characters in the password.

So a strict password policy will look something like the below.

password requisite pam_cracklib.so try_first_pass retry=2 minlength=15 lcredit=2 ucredit=3 dcredit=3 ocredit=4 difok=3

So if you have something like the above shown complexity policy in the file **/etc/pam.d /system-auth,** you can remove them if you want to get rid of them. But if you look from a security point of view they are a good method that can significantly increase the security of the system.

You can even disable the use of the module **pam_cracklib.so** by modifying the file **/etc/sysconfig/authconfig,** and setting **USECRACKLIB=no**

Hope this solves your issue, and was helpful in understanding password complexity settings in Linux.

reply (/comment/reply/173/77)

Thank you sir for your reply. (/comment/111#comment-111)

Permalink (/comment/111#comment-111) Submitted by Siddharth on Wed, 07/10/2013 - 20:59

Thank you sir for your reply.
that solved my problem and i learn new thing..

reply (/comment/reply/173/111)

Nice Articles for Beginners (/comment/97#comment-97)

Permalink (/comment/97#comment-97) Submitted by Indra Jeet Nagda on Sun, 06/30/2013 - 11:22

It is a really a good and detailed article written for the purpose of understanding the implementation of password mechanism in the linux system.

reply (/comment/reply/173/97)

Hi Indra Jeet, (/comment/98#comment-98)

Permalink (/comment/98#comment-98) Submitted by Sarath Pillai on Sun, 06/30/2013 - 15:22

Hi Indra Jeet,

Thanks for the comment and a warm welcome to slashroot.in.

Regards

reply (/comment/reply/173/98)

Brother, i like your articles (/comment/99#comment-99)

Permalink (/comment/99#comment-99) Submitted by Mohd atif khan on Mon, 07/01/2013 - 09:53

love to read the articles u post...and u know the reason? the reason is the esy to understand manner. i have a request. please write an artcle about windows passwords and hash.

reply (/comment/reply/173/99)

Hi mohd atif khan, (/comment/100#comment-100)

Permalink (/comment/100#comment-100) Submitted by Sarath Pillai on Mon, 07/01/2013 - 12:56

Hi mohd atif khan,

Good to know that you like our articles.
We will surly include article related to how are passwords stored in windows.

Regards

reply (/comment/reply/173/100)

hello (/comment/110#comment-110)

Permalink (/comment/110#comment-110) Submitted by Siddharth on Wed, 07/10/2013 - 20:54

hello
grt tutorial.

i have a question..
if i have salt(of an encrypted password) and hash(from which password is generated) can we generate original password from that..

thanks

reply (/comment/reply/173/110)

How to generate a shadow (/comment/139#comment-139)

Permalink (/comment/139#comment-139) Submitted by pradeep on Wed, 09/25/2013 - 10:41

How to generate a shadow style password hash for sha512?

reply (/comment/reply/173/139)

How to generate a shadow (/comment/140#comment-140)

Permalink (/comment/140#comment-140) Submitted by jalakam on Wed, 09/25/2013 - 10:43

How to generate a shadow style password hash for sha512?

reply (/comment/reply/173/140)

Hi Pradeep & jalkam, (/comment/532#comment-532)

Permalink (/comment/532#comment-532) Submitted by Sarath Pillai on Sat, 01/11/2014 - 03:21

Hi Pradeep & jalkam,

Creating sha512 type password for shadow files in debian systems can be done by using the below tool..

mkpasswd -m sha-512

In centos/red hat try using

grub-crypt --sha512

I have not tried the grub-crypt method myseld. Let me know if that worked..

Regards

reply (/comment/reply/173/532)

grub-crypt (/comment/548#comment-548)

Permalink (/comment/548#comment-548) Submitted by wombat on Thu, 01/30/2014 - 20:39

Hi Sarath,
very nice article. The tool to generate the password in fedora/centos/red hat is

grub-crypt --sha-512

regards

reply (/comment/reply/173/548)

grub-crypt (/comment/549#comment-549)

Permalink (/comment/549#comment-549) Submitted by wombat on Thu, 01/30/2014 - 20:48

Hi Sarath,
very nice article. The tool to generate the password in fedora/centos/red hat is

grub-crypt --sha-512

regards

reply (/comment/reply/173/549)

cool (/comment/538#comment-538)

Permalink (/comment/538#comment-538) Submitted by rizaldi on Tue, 01/21/2014 - 07:53

thanks for the article

reply (/comment/reply/173/538)

Ubuntu only accepting OpenLdap passwords stored in crypt (/comment/558#comment-558)

Permalink (/comment/558#comment-558) Submitted by Thomas McNeill on Fri, 02/07/2014 - 07:38

Hi Sarath,

I really appreciate the effort you take in teaching. I am stuck on a related topic.

All of our users are held in OpenLDAP. When their passwords are stored in anything other than crypt, logins to our Ubuntu servers fail with an authentication error. When passwords stored in sha or md5, they work on our Red Hat and Solaris servers.

I'd love to get some idea as to how to get Ubuntu to accept passwords hashed in sha.

Thanks.

reply (/comment/reply/173/558)

Retrieve paasword at login time (/comment/584#comment-584)

Permalink (/comment/584#comment-584) Submitted by Pooja on Sat, 03/01/2014 - 11:23

Hi Sir,

Very nice article.

I am facing a problem.Please help me to solve it.
I want to retrieve original password in a variable given at login time to validate it against some rules.

Help me how can i get it.

reply (/comment/reply/173/584)

Hi Puja, (/comment/586#comment-586)

Permalink (/comment/586#comment-586) Submitted by Sarath Pillai on Sun, 03/02/2014 - 02:31

Hi Puja,

Can you elaborate your particular use case? Coz if you want to compare the user typed password, you can already do it with PAM (Plug-gable authentication Modules). You can enforce a lot of rules there...

Please let me know..

Regards
Sarath

reply (/comment/reply/173/586)

fun read (/comment/754#comment-754)

Permalink (/comment/754#comment-754) Submitted by newageserversx on Mon, 06/23/2014 - 22:06

that was fun to read...thanks

reply (/comment/reply/173/754)

its too nice (/comment/791#comment-791)

Permalink (/comment/791#comment-791) Submitted by Maulik Patel on Wed, 07/16/2014 - 10:54

its too nice
i got all my point in single page
thanks.

reply (/comment/reply/173/791)

random value - salt how is generated while the user enters pword (/comment/940#comment-940)

Permalink (/comment/940#comment-940) Submitted by vb on Mon, 10/20/2014 - 07:02

Hi,
good article. it would be great if you could clarify how the salt value is regenerated when the user

enters the password. it would need to be the same as the entry is allowed after comparing the two hashes.

reply (/comment/reply/173/940)

Helped me (/comment/1051#comment-1051)

Permalink (/comment/1051#comment-1051) Submitted by Sindhu on Thu, 02/19/2015 - 15:24

Thanks for the detailed article.

reply (/comment/reply/173/1051)

Much thanks (/comment/1074#comment-1074)

Permalink (/comment/1074#comment-1074) Submitted by Micmit007 on Fri, 04/24/2015 - 02:11

Great article. Thanks for taking the time to write this information down in a clear and understandable way.

reply (/comment/reply/173/1074)

10x (/comment/1164#comment-1164)

Permalink (/comment/1164#comment-1164) Submitted by Meyon on Mon, 08/17/2015 - 13:53

An awesome article!
10x for share bro.Keep up!

reply (/comment/reply/173/1164)

THANK YOU ! (/comment/1196#comment-1196)

Permalink (/comment/1196#comment-1196) Submitted by Anonymous on Mon, 09/21/2015 - 13:24

THANK YOU !
the info was really very helpful
keep updatin such info :)

reply (/comment/reply/173/1196)

Nice Article (/comment/2238#comment-2238)

Permalink (/comment/2238#comment-2238) Submitted by Ariv on Mon, 04/25/2016 - 17:29

Hi Really nice article.
Since it is distributed and anyone can modify the code right? Is it possible to modify the hash algorithm as per our wish?

reply (/comment/reply/173/2238)

sha512 scheme id is not showing in shadow (/comment/2323#comment-2323)

Permalink (/comment/2323#comment-2323) Submitted by Hitesh on Fri, 07/22/2016 - 10:39

My system is using sha512 encryption method for passwd encryption but in shadow file i couldn't see the scheme id $6 for sha512, even it has no scheme id at all.

Please let me know why and from which file does system get to know what encryption type needs to be used.

REgards,

reply (/comment/reply/173/2323)

a pinch of salt... (/comment/2699#comment-2699)

Permalink (/comment/2699#comment-2699) Submitted by CAPPA on Wed, 09/13/2017 - 03:01

Thanks again, for posting this article; it was informative and well laid out.

It also helped to demystify the password process using in the Linux/Unix world.

reply (/comment/reply/173/2699)

/etc/passwd file update (/comment/3289#comment-3289)

Permalink (/comment/3289#comment-3289) Submitted by lajwant on Tue, 05/29/2018 - 20:19

Great article..definitely cleared the topic

Just wanted to know if we manually edit /etc/passwd or /etc/shadow
how do we sync/update so that changes can be reflected.

reply (/comment/reply/173/3289)

Add new comment

Your name

Subject

Comment *

- No HTML tags allowed.
- Web page addresses and e-mail addresses turn into links automatically.
- Lines and paragraphs break automatically.

More information about text formats (/filter/tips)

CAPTCHA

This question is for testing whether or not you are a human visitor and to prevent automated spam submissions.

I'm not a robot

reCAPTCHA
Privacy - Terms

Save Preview

Search Articles ...

Today's Most Popular

(/iperf-how-test-network-speedperformancebandwidth)

IPERF: How to test network Speed,Performance,Bandwidth (/iperf-how-test-network-speedperformancebandwidth)

Archives (/taxonomy/term/2) - 21 comment(s) (https://www.slashroot.in/iperf-how-test-network-speedperformancebandwidth#comments#comments)

(/httphypertext-transfer-protocol-request-and-response)

HTTP(Hypertext Transfer Protocol) Request

and Response (/httphypertext-transfer-protocol-request-and-response)

Archives (/taxonomy/term/2) - 10 comment(s) (https://www.slashroot.in/httphypertext-transfer-protocol-request-and-response#comments#comments)

(/how-are-passwords-stored-linux-understanding-hashing-shadow-utils)

How are passwords stored in Linux (Understanding hashing with shadow utils) (/how-are-passwords-stored-linux-understanding-hashing-shadow-utils)

Security Notes (/security-notes) - 28 comment(s) (https://www.slashroot.in/how-are-passwords-stored-linux-understanding-hashing-shadow-utils#comments#comments)

(/difference-between-iterative-and-recursive-dns-query)

difference between iterative and recursive dns query (/difference-between-iterative-and-recursive-dns-query)

Archives (/taxonomy/term/2) - 44 comment(s) (https://www.slashroot.in/difference-between-iterative-and-recursive-dns-query#comments#comments)

Most Commented

(/how-does-traceroute-work-and-examples-using-traceroute-command)

How Does Traceroute Work and Example's of using traceroute command (/how-does-traceroute-work-and-examples-using-traceroute-command)

Networking (/networking) - 75 comment(s) (https://www.slashroot.in/how-does-traceroute-work-and-examples-using-traceroute-command#comments#comments)

(/san-vs-nas-difference-between-storage-area-network-and-network-attached-storage)

SAN vs NAS - Difference between a Storage Area Network and Network Attached Storage (/san-vs-nas-difference-between-storage-area-network-and-network-attached-storage)

Archives (/taxonomy/term/2) - 58 comment(s) (https://www.slashroot.in/san-vs-nas-difference-between-

Top Rated Articles

(/introduction-to-git-version-control-system)

Introduction to git version control system (/introduction-to-git-version-control-system)

Average: 5 (4 votes)

(/netstat-command-examples-and-its-usage)

Netstat command examples and its usage (/netstat-command-examples-and-its-usage)

Average: 5 (30 votes)

(/modify-your-swap-space-configuring-your-linux-machine-use-lvm-swap-space-0)

Modify your swap space by configuring

storage-area-network-and-network-attached-storage#comments#comments)

(/difference-between-iterative-and-recursive-dns-query)

difference between iterative and recursive dns query (/difference-between-iterative-and-recursive-dns-query)

Archives (/taxonomy/term/2) - 44 comment(s) (https://www.slashroot.in/difference-between-iterative-and-recursive-dns-query#comments#comments)

(/linux-booting-process-step-step-tutorial-understanding-linux-boot-sequence)

Linux Booting Process: A step by step tutorial for understanding Linux boot sequence (/linux-booting-process-step-step-tutorial-understanding-linux-boot-sequence)

Archives (/taxonomy/term/2) - 41 comment(s) (https://www.slashroot.in/linux-booting-process-step-step-tutorial-understanding-linux-boot-sequence#comments#comments)

your Linux machine to use LVM as Swap Space. (/modify-your-swap-space-configuring-your-linux-machine-use-lvm-swap-space-0)

Average: 5 (8 votes)

(/how-to-install-and-configure-git)

how to install and configure git (/how-to-install-and-configure-git)

Average: 5 (3 votes)

(/website-vulnerability-scanner)

website vulnerability scanner (/website-vulnerability-scanner)

Average: 5 (9 votes)

how to add an init script for nginx service (/how-add-init-script-nginx-service)

Average: 5 (6 votes)

(/software-raid-1-configuration-linux)

Software RAID 1 Configuration in Linux (/software-raid-1-configuration-linux)

Average: 5 (25 votes)

(/ifconfig-utility-linux-users)

ifconfig utility for linux users (/ifconfig-utility-linux-users)

Average: 5 (11 votes)

Jump back to navigation (#page)

Get in touch with The Authors

Sarath Pillai

Ph: +917303074400

Email: **sarath@slashroot.in** (mailto:sarath@slashroot.in)

Follow Us

Subscribe to our RSS Feed (rss.xml)

Follow us on Twitter (https://twitter.com/slashrootin)

Be a fan on Facebook (http://www.facebook.com/pages

/Slashrootin-A-technical-Blog/101642306662299)

Satish Tiwary

Ph: +919509452488

Email: **satish@slashroot.in** (mailto:satish@slashroot.in)

Recent Posts

(/how-delete-git-branch-locally-and-remotely) **How To Delete A Git Branch Locally And Remotely** (/how-delete-git-branch-locally-and-remotely)

Archives (/taxonomy/term/2) - *8 months 3 weeks* ago

(/difference-between-bashrc-and-bashprofile) **Difference Between .bashrc and .bash_profile** (/difference-between-bashrc-and-bashprofile)

Archives (/taxonomy/term/2) - *8 months 4 weeks* ago

(/difference-between-process-and-thread-linux) **Difference Between Process And Thread in Linux** (/difference-between-process-and-thread-linux)

Archives (/taxonomy/term/2) - *9 months 6 days* ago

(/how-does-ssltls-chain-certificates-and-its-validation-work) **How Does SSL/TLS Chain Certificates and Its Validation work?** (/how-does-ssltls-chain-certificates-and-its-validation-work)

Security Notes (/security-notes) - *9 months 2 weeks* ago

Last Viewed

(/backup-and-restore-router-configuration-file-using-tftp-server-packet-tracer-cisco-ccna) **Backup and Restore router configuration file using TFTP server with Packet Tracer -Cisco CCNA** (/backup-and-restore-router-configuration-file-using-tftp-server-packet-tracer-cisco-ccna)

Networking (/networking) - last view *4 sec* ago

(/fingerprinting-detect-remote-operating-system) **fingerprinting-detect remote operating system** (/fingerprinting-detect-remote-operating-system)

Security Notes (/security-notes) - last view *6 sec* ago

(/slowloris-http-dosdenial-serviceattack-and-prevention) **SLOWLORIS: HTTP DOS(Denial Of Service)attack and prevention** (/slowloris-http-dosdenial-serviceattack-and-prevention)

Security Notes (/security-notes) - last view *7 sec* ago

(/disown-command-linux-explained-example-usage) **disown command in Linux explained with Example usage** (/disown-command-linux-explained-example-usage)

Archives (/taxonomy/term/2) - last view *19 sec* ago