Linux Security Report

Single user mode exploit in centos 6

De Montfort University

**Contents**

Abstract …………………………….……………………………………………………………………………….. Page 2

Introduction ………………………………………………………………………………………………………….. Page 2

The Exploit ………………………………………………………………………………………………………….. Page 2

The Remediation Strategy ……………………………………………………………………………………… Page 5

Conclusion …………………………………………………………………………………………………………… Page 6

Recommendation …………………………………………………………………………………………………. Page 6

References …………………………………………………………………………………………………………… Page 8

**Linux Security Report**

Abstract:

The purpose of the investigation was to find a vulnerable Linux operating system with a working exploit. The exploit found was on CentOS 6 and allowed any user to change the boot settings within grub, disable the security module, SELinux, and ultimately change the root password for the system without any prior knowledge of the previous/current password. The investigation discovers that the exploit is quick and easy to accomplish with possibly disastrous results if carried out by the wrong person. The investigation found that the most appropriate way of dealing with the problem was to modify the init file so that the user wasn’t given a super user shell on login but instead asked to provide a root password before access is allowed.

Introduction:

Linux is regarded as one of the most secure operating systems available. The operating system was developed with an emphasis on security to be built in to the Linux kernel from the start, this is contrary to Windows operating system where they add security as a bolt on package to its core. Even though Linux is a secure operating system it will always have vulnerabilities, most get patched quickly with some being left unnoticed for a long period of time.

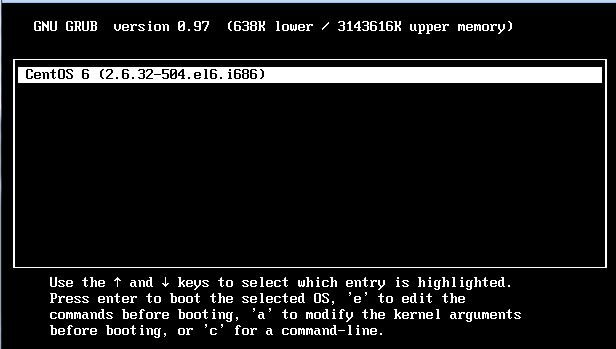
The vulnerability I will discuss in this report allows the user to reset the root password in single user mode without having any knowledge of the current root password. This is possible as when the user goes into single user mode they are given a root terminal. From this an unauthorised user can gain root access to the whole system and cause unprecedented damage. This is exploitable in CentOS 6.6, which uses Linux kernel 2.6 and the original init system.

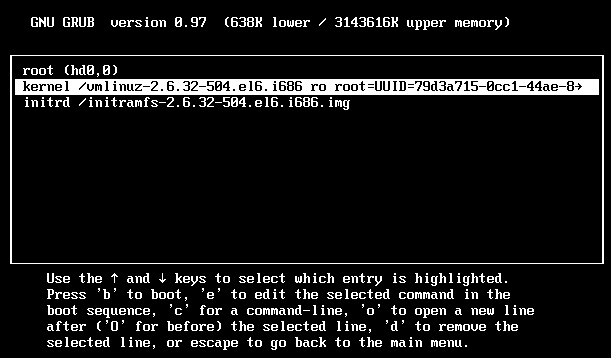
An exploit like this exists on Linux as it can be used innocently if the user forgets their root password to log-on to their machine. This is seen in many of the Linux distributions, however this exploit can be used by an unauthorised user to gain unlawful access to another person’s computer. The user should not be given root access straight away, there should be an authentication method in place before the user can gain root privileges. The vulnerability in CentOS 6 is very easy to exploit, much easier than any other distribution.

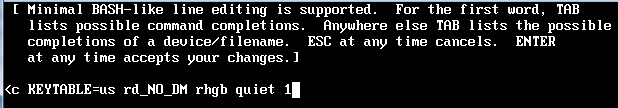
The Exploit:

The exploit in CentOS 6 is simple to carry out but a very effective vulnerability. This exploit allows the user to disable SELinux, then change the root password of the system without any knowledge of the current root password. This exploit is accomplished by firstly booting into run-level 1. By booting into run-level 1 (single user mode) you bypass login and go directly to a root terminal. In run-level 1 you change the root password then boot into run-level 3 (full multi user mode) which has authentication control. Here you login as root entering the password that was created in run-level 1, and the user will now have root access to the system.

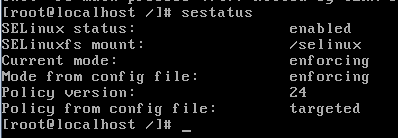
In the GRUB boot menu you force the system to boot to run-level 1. To do this you press ‘e’ at the first screen, this shows the operating system and the kernel that is used. The next page you navigate down one line to the kernel option and press ‘e’ again. Now at the end of the line add a space and add a number 1. Now press enter followed by ‘b’. This will now boot into run-level 1. This is possible as there is no control method set on the boot loader to stop changes to the booting of the system.





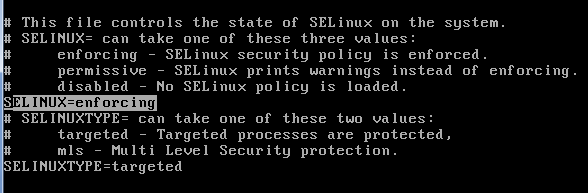


Once booted into run-level 1 you have to disable SELinux, the default security module of CentOS. Without disabling this module the exploit will not be successful. Firstly we need to check that SELinux is active, you do this by typing at the prompt *sestatus*, this results in the following;



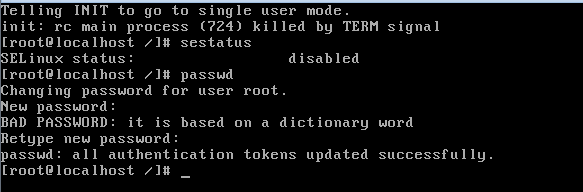
Here we see that SELinux is enabled and it is in enforcing mode. SELinux needs to be disabled permanently instead of just temporarily as after every reboot of the system SELinux will reset back to being active. The configuration file of SELinux is at */etc/sysconfig/selinux*.

Using the vim text editor you can open and modify the configuration file of SELinux so that it can be disabled.

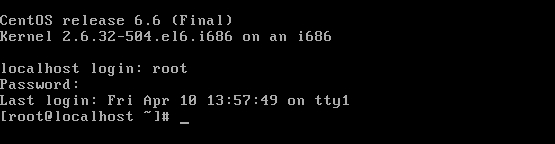


In the configuration file the word ‘enforcing’ needs to be replaced with ‘disabled’. Once this has been done and the change has been saved the system needs to be rebooted so that this change can take effect. After the reboot the system has no security module active and the exploit can be performed without any restrictions.

To change the root password in run-level 1 type the command *‘passwd’*, this will result in a prompt asking the user to type in a new root password. Here you can change the root password to anything you want and this will erase the current password that is set without any warnings.



After another reboot of the system, boot into run level 3 (full multi-user mode) and another terminal will appear which asks you to login. At this stage login to user account root and enter the new password that was created in run-level 1 and the exploit has been completed, a user now has total control over the system with their root account credentials.



The Remediation Strategy:

As seen by the discussed exploit, giving a user access to run-level 1 is a major security vulnerability. From here they are given a root terminal with allowances to change the settings of the operating system. So the solution to this exploit is to put password protection on run-level 1. This way if a user boots into run-level 1 the first thing that appears would be a prompt asking the user to enter the root password to continue.

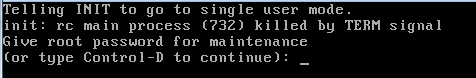
To put password protection on run-level 1 the operating systems init file needs to be modified. The file is located at */etc/sysconfig/init.* The init file “controls how the systems appears and functions during the boot process.” (centos.org, n.d.) Only one line of this file needs to be changed, this is the last line of the file that reads;



The above command says that should single user mode be logged on to, the user will get a super user capable shell, i.e. give that user a root terminal. That command needs to be changed to;



The above command now protects single user mode from unauthorized access as it now says that if anyone opens up single user mode they must provide a root password to continue. As seen in the screenshot below.



The remediation for this exploit is now complete, the source of the exploit is protected and therefore the system has been secured.

I believe the above solution to the problem is the most effective as it totally shuts out any unwanted user from accessing the single user mode. Single user mode is reserved primarily for system maintenance and clean up, therefore the only user who should have access to this run-level is the root user. The remediation plan I have implemented means that no-one other than root can perform system maintenance, this adds another layer to the security of the system.

You could argue that once the above solution has been implemented a user cannot ever change the password if they have genuinely forgotten it. This seems the case but there is another way in which the user could reset their password if needed. This can be done by editing the kernel in grub, just like the exploit, and adding *init=/bin/bash* instead of the number 1. This will now load the user into a bash 4.1 shell with no requirement for a password. Here you type:

*Mount –o remount,rw /*

By typing the above command into the bash shell you are remounting the root partition to a read-write partition. Originally root mount is read only. Now the root partition has been changed to have read-write permissions, you type *passwd* and now just like the discussed exploit you can change the root password. For this exploit I have recommended a solution as can be seen later in this report.

Conclusion:

The discussed vulnerability in CentOS 6 is a major one. A user should not be given root access without first authenticating themselves. The vulnerability contradicts some of the key aspects of computer security such as integrity, authorised changes and accountability. The user should have least privilege at all times, using the command *su to* elevate their privilege. By using *su,* there will be accountability of actions taken and most importantly a log file containing such information. There are a number of vulnerabilities in this operating system as I have discussed but if these problems are fixed properly then the system will be more secure and safe for use in a corporate environment.

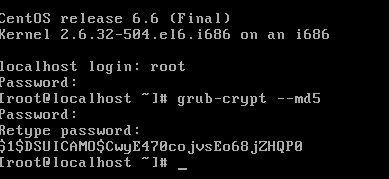
Recommendation:

To further secure CentOS 6 I would recommend putting a password lock on the grub boot loader. By doing this it will prevent any unauthorised user from editing the kernel before system boot. I would advise that the password set on the grub boot loader is different from any other passwords used on the system. If a new password is used to protect the boot loader from any changes then it all helps to add layers of security to the system and ensure that is not compromised.

To put a password lock on the grub boot loader firstly type:

*grub-crypt –md5*

Then enter a password, once done, an md5 hash will be generated from the given password.



Now the md5 hash has been generated that hash needs to be copied and added to the grub.conf file. In this file it contains all of the configuration information that is needed by the boot loader in order to function correctly.

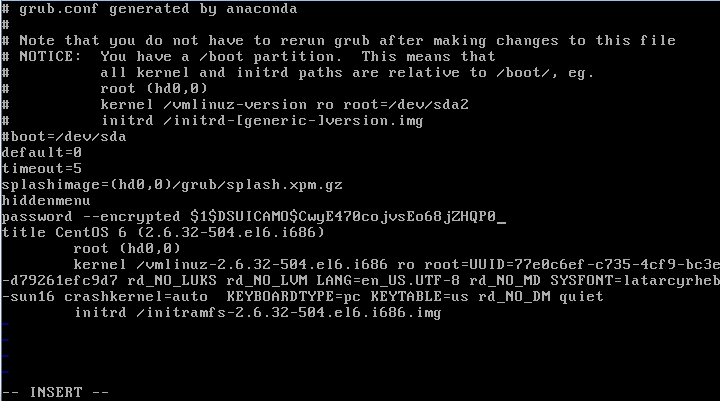
To edit the grub.conf file enter at the prompt:

Vi /etc/grub.conf

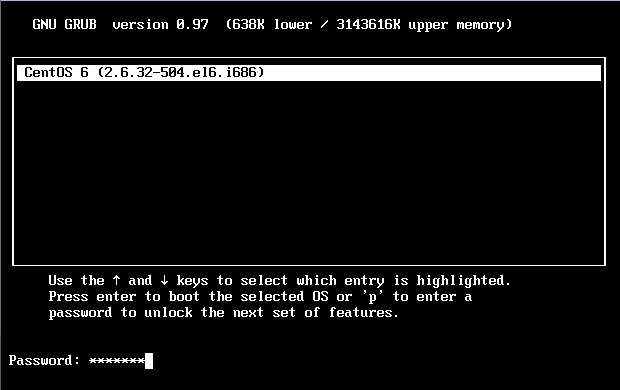
Now about half way down the page, add a line after the ‘hiddenmenu’ line and add:

password --encrypted $1$DSUICAMO$CwyE470cojvsEo68jZHQP0

(Note - the generated hash in this example will be different if this activity is repeated. If it is repeated then replace the hash in this example with your own hash)



Save the file and reboot the system. After the reboot the grub boot menu will appear normal however if you try and edit the options on the menu there will be no affect. You have to press ’p’ which will then allow you to enter your password for the grub boot loader. Once the password is entered you can edit the kernel options as discussed earlier in this report.



References:

centos.org, n.d. *28.1.14. /etc/sysconfig/init.* [Online]   
Available at: https://www.centos.org/docs/5/html/5.2/Deployment\_Guide/s2-sysconfig-init.html  
[Accessed April 2015].

Good linux tips, n.d. *How to password protect the grub boot loader.* [Online]   
Available at: http://www.good-linux-tips.com/2014/07/how-to-password-protect-grub-boot-loader.html  
[Accessed April 2015].

putorius.net, n.d. *Password protect single user mode in RedHat or Cent OS.* [Online]   
Available at: http://www.putorius.net/2013/07/password-protect-single-user-mode-in.html  
[Accessed April 2015].

tecmint.com, n.d. *How to hack your own linux system.* [Online]   
Available at: http://www.tecmint.com/how-to-hack-your-own-linux-system/  
[Accessed April 2015].