#!/bin/bash

# Linux Hardening Script for Ubuntu Server 20.04 LTS

##############################################################################################################

banner(){

echo " Welcome to Ubuntu Server 20.04 LTS Hardening Process "

echo

}

########################################################################################

# Check if running with root User

clear

banner

check\_root() {

if [ "$USER" != "root" ]; then

echo "Permission Denied"

echo "Can only be run by root"

exit

else

echo "you are running this script as $USER"

If [ $? -eq 0 ]; then

echo “ your command has been executed successfully”

fi

fi

}

########################################################################################

# Disable Unused Filesystems

disable\_unused\_fs(){

clear

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo -e "\e[93m[+]\e[00m Disabling Unused File Systems"

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo ""

for i in cramfs freevxfs jffs2 hfs hfsplus udf vfat usb-storage dccp sctp rds tipc ;

do

touch /etc/modprobe.d/$i.conf && echo "install $i /bin/true" > /etc/modprobe.d/$i.conf

done

if [ $? -eq 0 ] ; then

echo "your command has been executed successfully"

else

echo "your command is not executed successfully"

fi

}

# Ensure Mounting and FS Options Are configured Properly

set\_mounting\_options(){

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo -e "\e[93m[+]\e[00m Setting Mounting Options on /tmp and /dev/shm filesystems"

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo ""

for i in nodev nosuid noexec ;

do

mount | grep -E '\s/tmp\s' | grep $i >> /dev/null

if [ $? -eq 0 ]

then

echo "$i is already set on /tmp filesystem"

else

mount -o remount,$i /tmp

fi

done

for i in nodev nosuid noexec ;

do

mount | grep -E '\s/dev/shm\s' | grep $i >> /dev/null

if [ $? -eq 0 ]

then

echo "$i is already set on /dev/shm filesystem"

else

mount -o remount,$i /dev/shm

fi

done

if [ $? -eq 0 ] ; then

echo "your command has been executed successfully"

else

echo "your command is not executed successfully"

fi

}

# Set the sticky bit on all world writable directories:

set\_stickybit(){

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo -e "\e[93m[+]\e[00m Setting Sticky Bit on all world writable directories"

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo ""

df --local -P | awk '{if (NR!=1) print $6}' | xargs -I '{}' find '{}' -xdev -type d \( -perm -0002 -a ! -perm -1000 \) 2>/dev/null | xargs -I '{}' chmod a+t '{}'

if [ $? -eq 0 ] ; then

echo "your command has been executed successfully"

else

echo "your command is not executed successfully"

fi

}

# Disable Automounting

disable\_automounting(){

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo -e "\e[93m[+]\e[00m Disable Automounting "

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo ""

systemctl --now mask autofs

if [ $? -eq 0 ] ; then

echo "your command has been executed successfully"

else

echo "your command is not executed successfully"

fi

}

# FileSystems Integrity Check. Ensure AIDE is installed and filesystems integrity is regularly checked

fs\_integrity\_check(){

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo -e "\e[93m[+]\e[00m Installing AIDE and Setting Cron for FS Integrity Check regularly "

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo ""

apt install aide aide-common && aideinit && mv /var/lib/aide/aide.db.new /var/lib/aide/aide.db

echo "0 5 \* \* \* /usr/bin/aide.wrapper --config /etc/aide/aide.conf --check" >> /var/spool/cron/crontabs/root

if [ $? -eq 0 ] ; then

echo "your command has been executed successfully"

else

echo "your command is not executed successfully"

fi

}

# Ensure AppArmor is installed and its profiles are in enforce or complain mode

apparmor(){

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo -e "\e[93m[+]\e[00m Installing AppArmor and Setting Its Profile in enforce mode "

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo ""

apt install apparmor && aa-enforce /etc/apparmor.d/\*

if [ $? -eq 0 ] ; then

echo "your command has been executed successfully"

else

echo "your command is not executed successfully"

fi

}

# Warning Banners

login\_local\_remote\_banner(){

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo -e "\e[93m[+]\e[00m Setting up Local and Remote Login Warning Msgs "

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo ""

# check motd file is present or not

file="/etc/motd"

if [ -f “$file” ] ; then

echo “file $file is present

echo “removing file $file permanently” && rm -fr $file

else

echo “ file $file is not present”

fi

echo “moving ahead with setting up local and remote banner”

echo “ setting local login msg”

echo "Authorized uses only. All activity may be monitored and reported." > /etc/issue && chown root:root /etc/issue && chmod u-x,go-wx /etc/issue

echo “ setting remote login msg”

echo "Authorized uses only. All activity may be monitored and reported." > /etc/issue.net && chown root:root /etc/issue.net && chmod u-x,go-wx /etc/issue.net

if [ $? -eq 0 ] ; then

echo "your command has been executed successfully"

else

echo "your command is not executed successfully"

fi

}

# Ensure prelink & GDM are removed

prelink\_gdm(){

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo -e "\e[93m[+]\e[00m Purging Prelink and Gdm Packages "

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo ""

echo “ purging prelink and gdm “

apt purge prelink && apt purge gdm3

if [ $? -eq 0 ] ; then

echo "your command has been executed successfully"

else

echo "your command is not executed successfully"

fi

}

# Ensure xinetd, openbsd-inetd, X Window System, Avahi Server, CUPS, dhcp, ldap,nfs,dns,ftp, imap & pop3,samba,snmp,rsync,nis & http squid server are not installed

network\_services(){

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo -e "\e[93m[+]\e[00m Purging Unused Network Services and its Client Services "

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo ""

for i in xinetd openbsd-inetd xserver-xorg\* avahi-daemon cups isc-dhcp-server slapd rpcbind bind9 vsftpd dovecot-imapd dovecot-pop3d samba squid snmpd rsync nis rsh-client talk telnet ldap-utils

do

echo “ $i service, we are purging it now”

echo “ Purging $i service “

apt purge $i;done

done

if [ $? -eq 0 ] ; then

echo "your command has been executed successfully"

else

echo "your command is not executed successfully"

fi

}

# Setup Network Configuration Kernel Parameters: Ensure packet redirect sending disabled, ip forwarding disabled, source routed packets disabled,icmp redirects disabled, secure icmp redirect disabled,suspicious packets are logged, broadcast icmp request disabled, bogus icmp responses ignored,reverse path filtering enabled, tcp sync cookies enabled, ipv6 router disabled (Automated)

network\_conf\_sysctl(){

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo -e "\e[93m[+]\e[00m Changing Kernel Parameters For Network Configuration"

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo ""

for netconf in `cat /root/scripts/netconf.txt`

do

echo "$netconf" >> /etc/sysctl.conf && sysctl -w $netconf

done

if [ $? -eq 0 ] ; then

echo "your command has been executed successfully"

else

echo "your command is not executed successfully"

fi

}

# Firewall Configuration : Installation & Configuration of UFW

ufw\_firewall(){

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo -e "\e[93m[+]\e[00m UFW Firewall Installation & Configuration"

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo ""

echo “ Installing ufw firewall “ && apt install ufw

# Ensure loopback traffic is configured (Automated)

ufw allow in on lo && ufw deny in from 127.0.0.0/8 && ufw deny in from ::1 && ufw allow out on all

if [ $? -eq 0 ] ; then

echo "your command has been executed successfully"

else

echo "your command is not executed successfully"

fi

}

# Logging & Auditing. # Ensure auditd is installed & service is enabled. #Ensure journald is configured to send logs to rsyslog, compress large log files & write logfiles to persistent disk Edit the /etc/systemd/journald.conf file and add the following lines:

audit\_rsyslog\_journald(){

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo -e "\e[93m[+]\e[00m Installation and Configuration of Auditd, Rsyslog & Journald "

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo ""

apt install auditd audispd-plugins && systemctl --now enable auditd

apt install rsyslog && systemctl --now enable rsyslog && echo "$FileCreateMode 0640" >> /etc/rsyslog.conf

echo "ForwardToSyslog=yes" >> /etc/systemd/journald.conf && echo "Compress=yes" >> /etc/systemd/journald.conf && echo "Storage=persistent" >> /etc/systemd/journald.conf

if [ $? -eq 0 ] ; then

echo "your command has been executed successfully"

else

echo "your command is not executed successfully"

fi

}

# Ensure cron daemon is enabled and running. set permission on crontab files

cron(){

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo -e "\e[93m[+]\e[00m Ensure Cron is Enabled,Running & Set Permission on Crontab Files "

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo ""

systemctl --now enable cron && chown root:root /etc/crontab && chmod og-rwx /etc/crontab

if [ $? -eq 0 ] ; then

echo "your command has been executed successfully"

else

echo "your command is not executed successfully"

fi

}

# sshd configuration

ssh(){

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo -e "\e[93m[+]\e[00m Ensure Cron is Enabled,Running & Set Permission on Crontab Files "

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo ""

chown root:root /etc/ssh/sshd\_config && chmod og-rwx /etc/ssh/sshd\_config

sed -i '/X11Forwarding/s/^/#/g' /etc/ssh/sshd\_config

cat << EOF >> /etc/ssh/sshd\_config

LogLevel INFO

X11Forwarding no

MaxAuthTries 4

IgnoreRhosts yes

HostbasedAuthentication no

PermitRootLogin no

PermitEmptyPasswords no

PermitUserEnvironment no

Ciphers chacha20-poly1305@openssh.com,aes256-gcm@openssh.com,aes128-gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-ctr

MACs hmac-sha2-512-etm@openssh.com,hmac-sha2-256-etm@openssh.com,hmac-sha2-512,hmac-sha2-256

KexAlgorithms curve25519-sha256,curve25519-sha256@libssh.org,diffie-hellman-group14-sha256,diffie-hellman-group16-sha512,diffie-hellman-group18-sha512,ecdh-sha2-nistp521,ecdh-sha2-nistp384,ecdh-sha2-nistp256,diffie-hellman-group-exchange-sha256

ClientAliveInterval 300

ClientAliveCountMax 3

Banner /etc/issue.net

AllowTcpForwarding no

maxstartups 10:30:100

MaxSessions 10

EOF

systemctl restart sshd

if [ $? -eq 0 ] ; then

echo "your command has been executed successfully"

else

echo "your command is not executed successfully"

fi

}

# Setting File Permissions

file\_permission(){

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo -e "\e[93m[+]\e[00m Setting File Permission on Critical System Files "

echo -e "\e[34m---------------------------------------------------------------------------------------------------------\e[00m"

echo ""

chmod -R g-wx,o-rwx /var/log/\*

chown root:root /etc/ssh/sshd\_config

chmod og-rwx /etc/ssh/sshd\_config

chown root:root /etc/passwd

chmod 644 /etc/passwd

chown root:shadow /etc/shadow

chmod o-rwx,g-wx /etc/shadow

chown root:root /etc/group

chmod 644 /etc/group

chown root:shadow /etc/gshadow

chmod o-rwx,g-rw /etc/gshadow

chown root:root /etc/passwd-

chmod 600 /etc/passwd-

chown root:root /etc/shadow-

chmod 600 /etc/shadow-

chown root:root /etc/group-

chmod 600 /etc/group-

chown root:root /etc/gshadow-

chmod 600 /etc/gshadow-

if [ $? -eq 0 ] ; then

echo "your command has been executed successfully"

else

echo "your command is not executed successfully"

fi

}

function menu {

clear

echo

echo -e "\t\t\tHardening Menu\n"

echo -e "\t1. Check Script is running using root user"

echo -e "\t2. Disable Unused FileSystems"

echo -e "\t3. Set Mounting Options on /tmp and /dev/shm filesystems"

echo -e “\t4. Setting Sticky Bit on all world writable directories”

echo -e “\t5. Disable Automounting”

echo -e “\t6. Install AIDE and Set Cron for FS Integrity Check regularly”

echo -e “\t7. Install AppArmor and Set its profile in enforce mode”

echo -e “\t8. Setup Local and Remote Login Msgs”

echo -e “\t9. Purge Prelink and Gdm3 Package”

echo -e “\t10. Purge Unused Network Services and Its Client Services”

echo -e “\t11. Change Kernel Parameters For Network Configuration”

echo -e “\t12. Installation of UFW Firewall and Configuration”

echo -e "\t13.Installation and Configuration Auditd, Rsyslog & Journald"

echo -e "\t14.Ensure Cron is Enabled, Running and Set Permission on Crontab Files"

echo -e "\t15.SSH Configuration"

echo -e "\t16.Set File Permission On System Critical Files"

echo -e "\t0. Exit Menu\n\n"

echo -en "\t\tEnter an Option: "

read -n 100 option

}

while [ 100 ]

do

menu

case $option in

0)

break ;;

1)

check\_root ;;

2)

disable\_unused\_fs ;;

3)

set\_mounting\_options ;;

4)

set\_stickybit ;;

5)

disable\_automounting ;;

6)

fs\_integrity\_check ;;

7)

Apparmor ;;

8)

login\_local\_remote\_banner ;;

9)

prelink\_gdm ;;

10)

network\_services ;;

11)

network\_conf\_sysctl ;;

12)

ufw\_firewall ;;

13)

audit\_rsyslog\_journald ;;

14)

cron ;;

\*)

15)

ssh ;;

16)

file\_permission ;;

clear

echo "Sorry, wrong selection";;

esac

echo -en "\n\n\t\t\tHit any key to continue"

read -n 1 line

done

clear