GCDSTE
ENSA-M

Université Cadi Ayyad
Ecole Nationale Des Sciences Appliqués
Ministère De L'enseignement Supérieur, De La Recherche Scientifique Et De L'innovation Marrakech

MARRAKECH

PROJECT: LINUX HARDENING BEST PRACTICES AUTOMATION SCRIPT

Made by:
Youness Outaik
Walid Laanibi

Supervised by :
Mr. Omar Achbarou

2023/2024

SUMMARY

1. Introduction

1.1 Background

1.2 Objectives

2. Overview of ANSSI Guidelines

3. Automation Script Development

3.1 Script Architecture & Implementation

3.2 Interactive Features

4. Results

5. Challenges and Limitations

6. Conclusion

7. References

1. Introduction

In the dynamic landscape of information technology, the security of systems and networks stands as a paramount concern for organizations worldwide.
The evolving threat landscape necessitates the implementation of robust security measures to safeguard sensitive information, ensure business continuity, and maintain the trust of stakeholders.
As organizations increasingly rely on digital platforms for their operations, the need for effective security practices becomes more pressing than ever.

1.1 Background

The field of Information Security encompasses various strategies and best practices aimed at fortifying the resilience of systems against cyber threats. One prominent authority in this domain is the French National Cybersecurity Agency (ANSSI), which provides comprehensive guidelines and best practices for securing information systems.

These guidelines cover diverse aspects, from system hardening to user authentication, forming a valuable resource for organizations seeking to enhance their cybersecurity posture.

1.2 Objectives

The primary objective of this project is to automate the implementation of ANSSI's best practices and norms, focusing on specific chapters that address system hardening and Linux identification/authentication.

By developing a comprehensive shell script, this project aims to facilitate the integration of ANSSI's guidelines into Linux-based systems in an efficient and user-friendly manner.

The automation script will offer users the flexibility to select and execute specific hardening measures, aligning with ANSSI recommandations.

Through this automation effort, the project seeks to streamline the implementation of cybersecurity measures, thereby contributing to the overall security and resilience of information systems within an enterprise context.

2. Overview of ANSSI Guidelines

The ANSSI guidelines for Linux hardening provide a comprehensive framework to fortify the security of Linux-based operating systems, addressing vulnerabilities and mitigating potential risks. These guidelines, rooted in industry best practices, aim to enhance the resilience of information systems against evolving cyber threats.

Below is an overview of key considerations outlined in ANSSI's recommendations for securing Linux environments:

2.1. User Authentication and Authorization:

- ANSSI emphasizes the implementation of robust user authentication mechanisms, including the use of strong passwords and multi-factor authentication where applicable.

- Authorization policies are established to limit user privileges and ensure the principle of least privilege, reducing the risk of unauthorized access.

2.2. System Updates and Patch Management:

- Regular and timely updates are essential to address known vulnerabilities. ANSSI advocates for the systematic application of security patches to keep Linux systems up-to-date and resilient against known exploits.

2.3. Network Security:

- The guidelines provide recommendations for securing network services and configurations, including the use of firewalls and intrusion detection systems.
- Network services that are not essential to system functionality are advised to be disabled or restricted to minimize potential attack vectors.

2.4. File System Security:

- ANSSI suggests secure file system configurations, such as employing mandatory access controls (MAC) and implementing file integrity monitoring.
- Recommendations include securing sensitive system files and directories, and monitoring changes to critical files.

2.5. Logging and Auditing:

- Robust logging and auditing practices are encouraged to facilitate the detection and response to security incidents.

- ANSSI outlines guidelines for configuring audits, the Linux audit framework, to capture relevant security events and maintain comprehensive logs.

2.6. Secure Communication:

- Secure communication channels, particularly in network services like SSH, are highlighted. ANSSI recommends the use of encryption protocols to protect sensitive data during transit.

2.7. Secure Boot and Kernel Hardening:

- ANSSI provides guidance on secure boot configurations and kernel hardening measures to prevent unauthorized modifications and enhance the overall integrity of the Linux system.

2.8. Application Security:

- Recommendations extend to securing applications running on Linux systems, emphasizing secure coding practices, regular code reviews, and vulnerability assessments.

2.9. Incident Response Planning:

- The guidelines advocate for the development and regular testing of incident response plans. Organizations are encouraged to establish procedures to efficiently respond to security incidents.

2.10. Security Awareness and Training:

- ANSSI emphasizes the importance of fostering a security-aware culture within organizations. Regular training and awareness

programs are recommended to empower users to recognize and respond to security threats.

Incorporating these guidelines into Linux environments contributes to a more resilient and secure information system. The subsequent sections of this report will explore the methodology and implementation details of automating ANSSI's Linux hardening best practices through a shell script.

3. Automation Script Development

3.1 Script Architecture & Implementation

The script architecture is designed to provide a modular and organized framework that aligns with ANSSI's categorization of Linux hardening best practices. The script is structured into distinct sections, each dedicated to specific aspects of hardening.

This modular approach enhances readability, maintainability, and allows users to selectively execute hardening measures. The key components of the script architecture include:

a) User Authentication and Authorization

- Contains commands and configurations related to strengthening user authentication, enforcing password policies, and managing user privileges.

b) System Updates and Patch Management

- Includes commands for updating the system, applying security patches, and configuring automatic update mechanisms.

c) Network Security

- Addresses configurations related to firewall settings, network service restrictions, and intrusion detection measures.

d) File System Security

- Encompasses commands for securing file systems, implementing mandatory access controls, and monitoring file integrity.

e) Logging and Auditing

- Integrates configurations for setting up robust logging mechanisms using auditd, capturing relevant security events.

f) Secure Communication

- Focuses on securing communication channels, particularly in SSH, through encryption protocols and secure configurations.

g) Secure Boot and Kernel Hardening

- Includes commands for configuring secure boot options and implementing measures to harden the Linux kernel.

h) Application Security

- Addresses guidelines for securing applications running on Linux systems, emphasizing secure coding practices and vulnerability assessments.

i) Incident Response Planning

- Outlines configurations and measures related to incident response planning and testing.

j) Security Awareness and Training

- Covers recommendations for fostering a security-aware culture within organizations through regular training and awareness programs.

3.2 Interactive Features

The script incorporates interactive features to enhance user engagement and flexibility during the hardening process. Interactive elements include:

a) User Selection Menu:

- Presents users with a menu allowing them to choose specific hardening sections or execute all sections collectively.

b) Parameter Input Prompts:

- Prompts users for necessary inputs, such as passwords, configuration settings, or preferences, allowing customization.

c) Confirmation Prompts:

- Seeks confirmation from users before implementing critical configurations to prevent unintended changes.

d) Progress Indicators:

- Provides clear progress indicators and status messages to keep users informed about the script's execution.

e) Error Handling and Guidance:

- Incorporates robust error-handling mechanisms with clear error messages and guidance to assist users in case of issues.

f) Logging and Reporting:

- Generates logs to record the actions performed by the script, offering users insight into changes made during the hardening process.

These interactive features aim to create a user-friendly experience, allowing users to tailor the hardening process based on their requirements and organizational policies. The next sections will delve into the testing, validation, and outcomes of the automation script.

4. Results

Name	Script
MainScript **Menu**	#menu while true; do clear echo "===== Menu =====" echo "0. Introduction" echo "1. ANSSI Hardware Hardening Script" echo "2. ANSSI Kernel Configuration Hardening Script" echo "3. ANSSI Disk Partition Hardening Script" echo "4. ANSSI Authentication and Identification Hardening Script" echo "5. ANSSI File Protection Hardening Script" echo "6. ANSSI Network Hardening Script" echo "7. Exit" echo "================" read -p "Enter your choice: " choice case $choice in 0) #Introduction echo "Introduction" echo "LINUX HARDENING AUTOMATION SCRIPT" echo "Made by OUTAIK Youness & LAANIBI Walid" echo "Under the supervision of teacher ACHBAROU Omar" ;; 1) #1. ANSSI Hardware Hardening Script echo "1. ANSSI Hardware Hardening Script"

```
    # Prompt the user
     read -p "Do you want to execute the bash script? (yes/no): " answer
     # Check the user's response
     if [[ "$answer" == "yes" ]]; then
     echo "Executing the script..."
          sudo ./L1.sh
     elif [[ "$answer" == "no" ]]; then
          echo "Not executing the script."
     else
          echo "Invalid input. Please enter 'yes' or 'no'."
     fi
    ;;

2)
    #2. ANSSI Kernel Configuration Hardening Script

    echo "2. ANSSI Kernel Configuration Hardening Script"
    # Prompt the user
     read -p "Do you want to execute the bash script? (yes/no): " answer
     # Check the user's response
     if [[ "$answer" == "yes" ]]; then
     echo "Executing the script..."
          sudo ./L2.sh
     elif [[ "$answer" == "no" ]]; then
          echo "Not executing the script."
     else
          echo "Invalid input. Please enter 'yes' or 'no'."
     fi
    ;;

3)
    #3. ANSSI Disk Partition Hardening Script

    echo "3. ANSSI Disk Partition Hardening Script"
```

```
        # Prompt the user
         read -p "Do you want to execute the bash script? (yes/no): " answer
         # Check the user's response
         if [[ "$answer" == "yes" ]]; then
         echo "Executing the script..."
              sudo ./L3.sh
         elif [[ "$answer" == "no" ]]; then
              echo "Not executing the script."
         else
              echo "Invalid input. Please enter 'yes' or 'no'."
         fi
        ;;

    4)
        #4. ANSSI Authentication and Identification Hardening Script

        echo "4. ANSSI Authentication and Identification Hardening Script"
        # Prompt the user
         read -p "Do you want to execute the bash script? (yes/no): " answer
         # Check the user's response
         if [[ "$answer" == "yes" ]]; then
         echo "Executing the script..."
              sudo ./L4.sh
         elif [[ "$answer" == "no" ]]; then
              echo "Not executing the script."
         else
              echo "Invalid input. Please enter 'yes' or 'no'."
         fi
        ;;

    5)
        #5. ANSSI File Protection Hardening Script

        echo "5. ANSSI File Protection Hardening Script"
```

```
    # Prompt the user
     read -p "Do you want to execute the bash script? (yes/no): " answer
     # Check the user's response
     if [[ "$answer" == "yes" ]]; then
     echo "Executing the script..."
          sudo ./L5.sh
     elif [[ "$answer" == "no" ]]; then
          echo "Not executing the script."
     else
          echo "Invalid input. Please enter 'yes' or 'no'."
     fi
    ;;

6)
    #6. ANSSI Network Hardening Script

    echo "6. ANSSI Network Hardening Script"
    # Prompt the user
     read -p "Do you want to execute the bash script? (yes/no): " answer
     # Check the user's response
     if [[ "$answer" == "yes" ]]; then
     echo "Executing the script..."
          sudo ./L6.sh
     elif [[ "$answer" == "no" ]]; then
          echo "Not executing the script."
     else
          echo "Invalid input. Please enter 'yes' or 'no'."
     fi
    ;;

7)
    #Exiting…

    echo "Exiting..."
```

	exit 0 ;; *) echo "Invalid choice. Please enter a number between 1 and 7." ;; esac read -n 1 -s -r -p "Press any key to continue..." done
L1	# Disable USB ports echo "blacklist usb-storage" > /etc/modprobe.d/disable-usb-storage.conf modprobe -r usb-storage # Restrict access to the serial ports chmod 700 /dev/ttyS* # Disable Firewire modules echo "install firewire-core /bin/true" > /etc/modprobe.d/disable-firewire.conf echo "install firewire-ohci /bin/true" >> /etc/modprobe.d/disable-firewire.conf # Restrict physical console access echo "console" >> /etc/securetty # Prompt the user to set a BIOS/UEFI password read -p "Do you want to set a BIOS/UEFI password? (yes/no): "

```
set_password

if [[ $set_password == "yes" ]]; then
    read -sp "Enter the BIOS/UEFI password: " bios_password



    echo "Setting BIOS/UEFI password to: $bios_password"



    echo "BIOS/UEFI password set successfully."
else
    echo "No BIOS/UEFI password set. Continuing with other hardening
measures."
fi


# Display BIOS/UEFI Hardening message

echo "ANSSI BIOS/UEFI Password Hardening measures applied
successfully."


 # Check if the script is being run as root

if [ "$EUID" -ne 0 ]; then
  echo "Please run this script as root."
  exit 1
fi

 # Backup the GRUB configuration file

cp /etc/default/grub /etc/default/grub.bak
```

```
 # Set the GRUB_DISABLE_RECOVERY option to restrict recovery

mode
sed -i
's/GRUB_DISABLE_RECOVERY="false"/GRUB_DISABLE_RECOVE
RY="true"/' /etc/default/grub

# Update GRUB configuration

update-grub

echo "Boot from external media has been disabled. GRUB
configuration updated."


# Check if the script is being run as root

if [ "$EUID" -ne 0 ]; then
  echo "Please run this script as root."
  exit 1
fi

# Check if the system is using UEFI

if [ ! -d /sys/firmware/efi ]; then
  echo "UEFI is not detected on this system. Secure Boot is not
applicable."
  exit 1
fi

# Check if Secure Boot is already enabled

if mokutil --sb-state | grep -q "SecureBoot enabled"; then
  echo "Secure Boot is already enabled."
  exit 0
fi
```

```
# Install necessary package for MOK (Machine Owner Key) management

apt-get install -y mokutil

# Enable Secure Boot

mokutil --enable-secure-boot

echo "Secure Boot has been enabled. Please follow the on-screen instructions to complete the process."


# Check if the script is being run as root

if [ "$EUID" -ne 0 ]; then
  echo "Please run this script as root."
  exit 1
fi

# Backup the GRUB configuration file

cp /etc/default/grub /etc/default/grub.bak

# Generate an encrypted password for GRUB

echo "Enter the desired GRUB password:"
read -s grub_password
encrypted_password=$(echo -e "$grub_password\n$grub_password" | grub-mkpasswd-pbkdf2 | grep "grub.pbkdf2.*" | awk '{print $NF}')

# Add the password to the GRUB configuration

sed -i "/^GRUB_CMDLINE_LINUX_DEFAULT=/ s/\"$/ GRUB_CMDLINE_LINUX_DEFAULT=\"quiet splash\"/"
```

	/etc/default/grub echo "GRUB_ENABLE_CRYPTODISK=y" >> /etc/default/grub echo "GRUB_PASSWORD=$encrypted_password" >> /etc/default/grub # Update GRUB configuration update-grub echo "GRUB password has been set. GRUB configuration updated." # Lock the root account passwd -l root # Set strong password policy sed -i 's/PASS_MAX_DAYS\t99999/PASS_MAX_DAYS\t90/' /etc/login.defs sed -i 's/PASS_MIN_DAYS\t0/PASS_MIN_DAYS\t7/' /etc/login.defs sed -i 's/PASS_WARN_AGE\t7/PASS_WARN_AGE\t14/' /etc/login.defs # Display a concluding message echo "ANSSI Hardware Hardening measures applied successfully."
L2	# Backup existing kernel configuration cp /boot/config-$(uname -r) /boot/config-$(uname -r).bak # Enable and configure address space layout randomization (ASLR) echo "CONFIG_RANDOMIZE_BASE=y" >> /usr/src/linux/.config echo "CONFIG_RANDOMIZE_MODULE_REGION_FULL=y" >>

	/usr/src/linux/.config # Set a maximum address space for mmap echo "CONFIG_DEFAULT_MMAP_MIN_ADDR=65536" >> /usr/src/linux/.config # Disable support for unnecessary file systems echo "CONFIG_SQUASHFS=n" >> /usr/src/linux/.config echo "CONFIG_UDF_FS=n" >> /usr/src/linux/.config echo "CONFIG_VFAT_FS=n" >> /usr/src/linux/.config make oldconfig # Restrict access to kernel logs chmod 600 /var/log/dmesg chmod 600 /var/log/kern.log # Enable process execution prevention echo "kernel.exec-shield = 1" >> /etc/sysctl.conf echo "kernel.randomize_va_space = 2" >> /etc/sysctl.conf sysctl -p # Restrict kernel module loading echo "install cramfs /bin/true" > /etc/modprobe.d/cramfs.conf echo "install freevxfs /bin/true" > /etc/modprobe.d/freevxfs.conf # Disable support for legacy 16-bit x86 code echo "CONFIG_X86_16BIT=y" >> /usr/src/linux/.config

```
# Restrict access to /dev/mem and /dev/kmem

# Check if the script is being run as root
if [ "$EUID" -ne 0 ]; then
  echo "Please run this script as root."
  exit 1
fi

echo "Enabling kernel hardening features..."

# Enable Address Space Layout Randomization (ASLR)

echo "kernel.randomize_va_space=2" >> /etc/sysctl.conf
sysctl -p

# Enable ExecShield

echo "kernel.exec-shield=1" >> /etc/sysctl.conf
sysctl -p

# Enable kernel module signing

echo "options modprobe modules.sig_enforce=1" >
/etc/modprobe.d/modules.conf

# Enable strict file permissions on kernel modules

chmod 0600 /etc/modprobe.d/modules.conf

# Enable kernel module loading/unloading restrictions

echo "install usb-storage /bin/true" >
/etc/modprobe.d/disable-usb-storage.conf
```

	# Enable stricter ptrace security echo "kernel.yama.ptrace_scope=3" >> /etc/sysctl.conf sysctl -p # Set core dumps to a secure location echo "fs.suid_dumpable = 0" >> /etc/sysctl.conf sysctl -p # Disable kernel support for some unsafe CPU features echo "install cramfs /bin/true" > /etc/modprobe.d/disable-cramfs.conf echo "install freevxfs /bin/true" > /etc/modprobe.d/disable-freevxfs.conf echo "install jffs2 /bin/true" > /etc/modprobe.d/disable-jffs2.conf echo "install hfs /bin/true" > /etc/modprobe.d/disable-hfs.conf echo "install hfsplus /bin/true" > /etc/modprobe.d/disable-hfsplus.conf # Restrict access to kernel logs chmod o-rwx /var/log/dmesg chmod o-rwx /var/log/kern.log echo "Kernel hardening features enabled." # Compile and install the new kernel make && make modules_install && make install # Display a concluding message echo "ANSSI Kernel Configuration Hardening applied successfully. Reboot the system to activate changes."
L3	# Ensure /tmp is on a separate partition echo "Creating a separate partition for /tmp..."

```
dd if=/dev/zero of=/tmp_partition bs=1M count=512
mkfs.ext4 /tmp_partition
mount -o loop /tmp_partition /tmp
echo "/tmp_partition /tmp ext4 loop 0 0" >> /etc/fstab

# Set noexec, nosuid, and nodev options for /tmp

echo "Setting noexec, nosuid, and nodev options for /tmp..."
mount -o remount,noexec,nosuid,nodev /tmp

# Set appropriate permissions for /var/tmp

echo "Setting appropriate permissions for /var/tmp..."
chmod 1777 /var/tmp

# Separate partitions for /home, /var, and /usr

echo "Creating separate partitions for /home, /var, and /usr..."

# We adjust the partition sizes based on our system and requirements

dd if=/dev/zero of=/home_partition bs=1M count=10240
mkfs.ext4 /home_partition
mount -o loop /home_partition /home
echo "/home_partition /home ext4 loop 0 0" >> /etc/fstab

dd if=/dev/zero of=/var_partition bs=1M count=20480
mkfs.ext4 /var_partition
mount -o loop /var_partition /var
echo "/var_partition /var ext4 loop 0 0" >> /etc/fstab

dd if=/dev/zero of=/usr_partition bs=1M count=40960
mkfs.ext4 /usr_partition
mount -o loop /usr_partition /usr
echo "/usr_partition /usr ext4 loop 0 0" >> /etc/fstab
```

	# Disable unnecessary mounts echo "Disabling unnecessary mounts..." echo "tmpfs /run/shm tmpfs defaults,noexec,nosuid,nodev 0 0" >> /etc/fstab # Display a concluding message echo "ANSSI Disk Partition Hardening measures applied successfully. Reboot the system to activate changes."
L4	# Set password policies echo "Configuring password policies..." echo "password requisite pam_pwquality.so retry=3 minlen=12 ucredit=-1 lcredit=-1 dcredit=-1 ocredit=-1" >> /etc/pam.d/common-password echo "auth required pam_tally2.so deny=5 unlock_time=1800" >> /etc/pam.d/common-auth # Enforce password complexity echo "Configuring password complexity..." apt-get install libpam-cracklib echo "password requisite pam_cracklib.so retry=3 minlen=12 ucredit=-1 lcredit=-1 dcredit=-1 ocredit=-1 difok=3" >> /etc/pam.d/common-password # Set account lockout policy echo "Configuring account lockout policy..." echo "auth required pam_tally2.so deny=5 unlock_time=1800" >> /etc/pam.d/common-auth

	# Disable root login echo "Disabling root login..." sed -i 's/PermitRootLogin yes/PermitRootLogin no/g' /etc/ssh/sshd_config # Limit access to the su command echo "Configuring su command restrictions..." echo "auth required pam_wheel.so" >> /etc/pam.d/su echo "wheel:x:10:username" >> /etc/group # Enable login banner echo "Configuring login banner..." echo "Authorized access only. All activity may be monitored and reported." > /etc/issue.net echo "Banner /etc/issue.net" >> /etc/ssh/sshd_config # Configure session timeout echo "Configuring session timeout..." echo "TMOUT=600" >> /etc/profile echo "readonly TMOUT" >> /etc/profile echo "export TMOUT" >> /etc/profile # Display a concluding message echo "ANSSI Authentication and Identification Hardening measures applied successfully. Restart services or reboot the system to activate changes."
L5	# Set restrictive permissions on sensitive files and directories echo "Setting restrictive permissions on sensitive files and directories..."

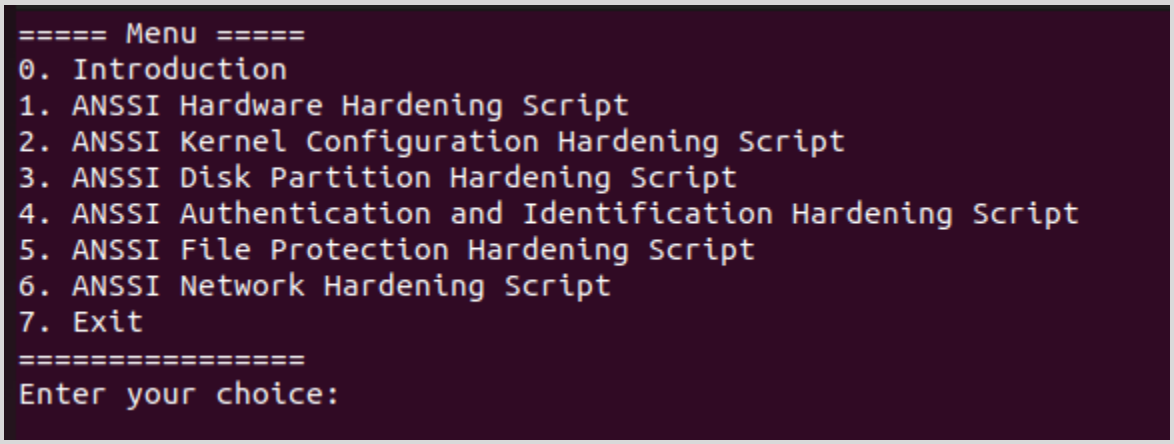
```
chmod 600 /etc/shadow
chmod 644 /etc/passwd
chmod 640 /etc/group
chmod 644 /etc/hosts.allow
chmod 644 /etc/hosts.deny
chmod 644 /etc/ssh/sshd_config
chmod 600 /etc/sudoers

# Restrict access to system logs

echo "Restricting access to system logs..."
chmod 640 /var/log/auth.log
chmod 640 /var/log/syslog
chmod 640 /var/log/messages

# Set proper ownership for critical files and directories

echo "Setting proper ownership for critical files and directories..."
chown root:root /etc/passwd
chown root:shadow /etc/shadow
chown root:root /etc/group
chown root:root /etc/hosts.allow
chown root:root /etc/hosts.deny
chown root:root /etc/ssh/sshd_config
chown root:root /etc/sudoers

# Set immutable attribute to critical files

echo "Setting immutable attribute to critical files..."
chattr +i /etc/passwd
chattr +i /etc/shadow
chattr +i /etc/group
chattr +i /etc/hosts.allow
chattr +i /etc/hosts.deny
chattr +i /etc/ssh/sshd_config
chattr +i /etc/sudoers
```

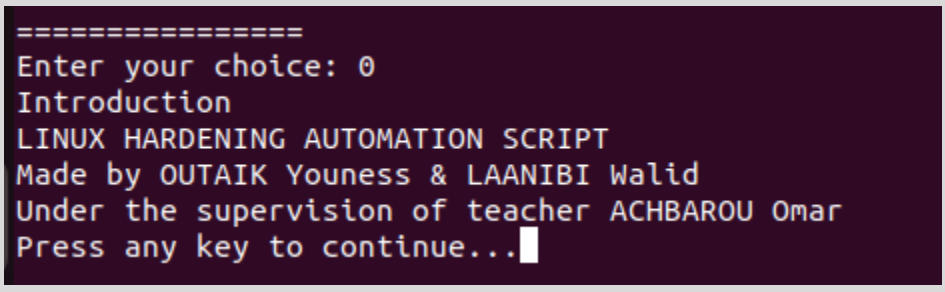
	# Secure sensitive directories echo "Securing sensitive directories..." chmod 700 /root chmod 750 /var/log # Disable world-writable permissions echo "Disabling world-writable permissions..." find / -type d -perm -002 -exec chmod o-w {} \; find / -type f -perm -002 -exec chmod o-w {} \; # Set SGID bit on critical directories echo "Setting SGID bit on critical directories..." chmod g+s /usr/bin chmod g+s /usr/sbin chmod g+s /bin chmod g+s /sbin # Display a concluding message echo "ANSSI File Protection Hardening measures applied successfully."
L6	# Disable unused network services echo "Disabling unused network services..." systemctl disable avahi-daemon systemctl disable cups systemctl disable nfs systemctl disable rpcbind systemctl disable postfix systemctl disable bluetooth systemctl disable apache2

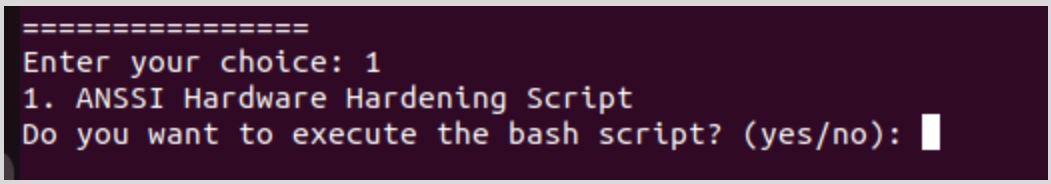
```
# Enable and configure firewall (iptables or nftables)

echo "Configuring firewall rules..."

# In our work, we allowed SSH and blocked everything else, it all depends on the rules you want to set:

iptables -P INPUT DROP
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
iptables -A INPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT

# Enable SYN flood protection

echo "Enabling SYN flood protection..."
sysctl -w net.ipv4.tcp_syncookies=1

# Disable IP forwarding

echo "Disabling IP forwarding..."
sysctl -w net.ipv4.ip_forward=0

# Disable ICMP Redirects

echo "Disabling ICMP Redirects..."
sysctl -w net.ipv4.conf.all.send_redirects=0
sysctl -w net.ipv4.conf.default.send_redirects=0

# Disable source routing

echo "Disabling source routing..."
sysctl -w net.ipv4.conf.all.accept_source_route=0
sysctl -w net.ipv4.conf.default.accept_source_route=0
```

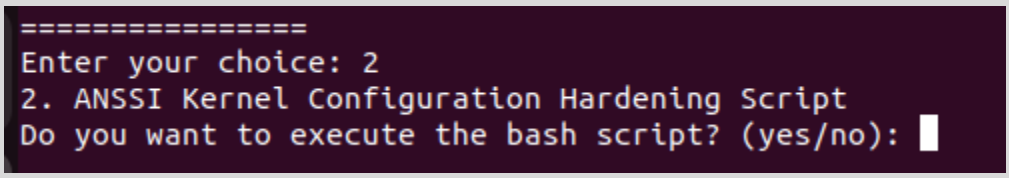
```
# Enable ARP protection

echo "Enabling ARP protection..."
sysctl -w net.ipv4.conf.all.arp_ignore=1
sysctl -w net.ipv4.conf.default.arp_ignore=1
sysctl -w net.ipv4.conf.all.arp_announce=2
sysctl -w net.ipv4.conf.default.arp_announce=2

# Disable IPv6 if not needed

echo "Disabling IPv6..."
echo "net.ipv6.conf.all.disable_ipv6 = 1" >> /etc/sysctl.conf
echo "net.ipv6.conf.default.disable_ipv6 = 1" >> /etc/sysctl.conf
sysctl -p

# Display a concluding message

echo "ANSSI Network Hardening measures applied successfully."
```

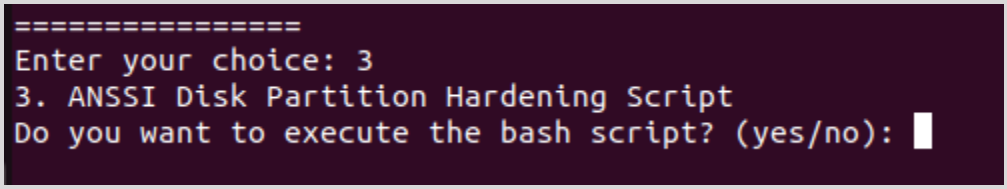
Results

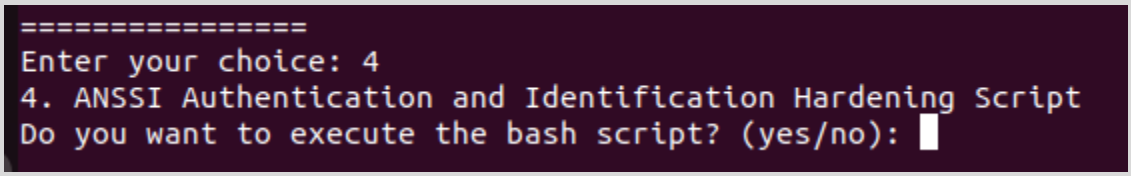


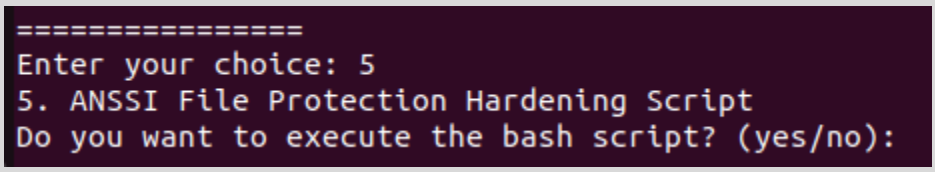


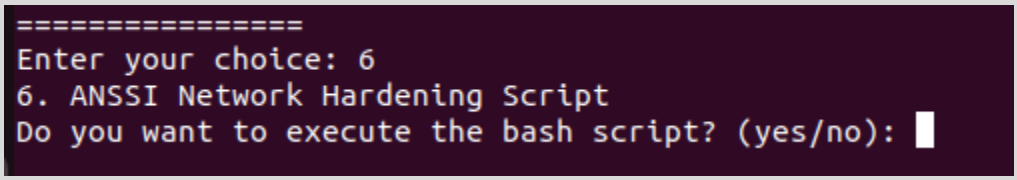

==========
Enter your choice: 1
1. ANSSI Hardware Hardening Script
Do you want to execute the bash script? (yes/no):

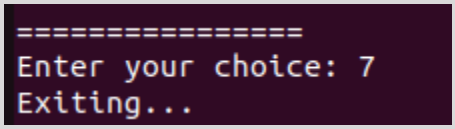

==========
Enter your choice: 2
2. ANSSI Kernel Configuration Hardening Script
Do you want to execute the bash script? (yes/no):


==========
Enter your choice: 3
3. ANSSI Disk Partition Hardening Script
Do you want to execute the bash script? (yes/no):


==========
Enter your choice: 4
4. ANSSI Authentication and Identification Hardening Script
Do you want to execute the bash script? (yes/no):


==========
Enter your choice: 5
5. ANSSI File Protection Hardening Script
Do you want to execute the bash script? (yes/no):


==========
Enter your choice: 6
6. ANSSI Network Hardening Script
Do you want to execute the bash script? (yes/no):


==========
Enter your choice: 7
Exiting...

5. Challenges and Limitations

5.1 Identified Challenges

During the development and implementation of the automation script for ANSSI Linux hardening guidelines, several challenges were encountered. Addressing these challenges was crucial to ensuring the effectiveness and reliability of the script.

5.1.1 Diverse Linux Distributions:

Adapting the script to accommodate the nuances of various Linux distributions posed a challenge. Differences in package management systems, file paths, and default configurations required careful consideration.

The script includes conditional statements to detect the Linux distribution and adjust commands accordingly. Extensive testing across diverse distributions was conducted to validate compatibility.

5.1.2 User Input Validation:

Validating user inputs, especially when configuring critical settings, required meticulous attention. Ensuring that users provide accurate and secure inputs without the risk of misconfigurations presented a challenge.

Implemented input validation checks within the script to verify the correctness and appropriateness of user inputs. Clear prompts and guidelines were provided to assist users in providing accurate information.

5.1.3 Customization Complexity:

Balancing the need for customization with the complexity of the script was challenging. Providing users with flexibility while maintaining script simplicity and coherence was a delicate balance.

The script includes a user-friendly menu system that guides users through the customization process. Extensive documentation accompanies the script to assist users in understanding and navigating the customization options.

5.1.4 Comprehensive Logging:

Capturing comprehensive logs while avoiding information overload was a challenge. Striking a balance between logging critical actions and maintaining log readability required thoughtful consideration.

The script was designed to log essential actions, errors, and configuration changes. Log entries were structured to provide meaningful insights without overwhelming users with excessive information.

5.2 Limitations

Despite efforts to address challenges, certain limitations exist within the current scope of the automation script for ANSSI Linux hardening guidelines.

5.2.1 Application-Specific Configurations:

The script focuses on general Linux hardening practices and may not cover application-specific configurations. Organizations with

specialized applications may need to supplement the script with additional configurations.

5.2.2 Continuous Monitoring:

The script provides hardening measures during execution, but continuous monitoring is essential for sustained security. Regular security audits and ongoing monitoring efforts are necessary to address emerging threats.

5.2.3 Dependency on User Inputs:

The effectiveness of the script relies on accurate user inputs. Inaccurate or incomplete information provided during script execution may result in misconfigurations. Clear user guidance is crucial to mitigate this limitation.

5.2.4 Limited Scope:

The script primarily focuses on ANSSI Linux hardening guidelines. Organizations with specific regulatory or compliance requirements may need to extend the script to cover additional standards.

5.2.5 One-Time Execution:

The script is designed for one-time execution, and subsequent changes to the system may require manual adjustments. Regular reviews and updates are recommended to adapt to evolving security landscapes.

5. Conclusion

Based on the ANSSI guidelines, our script stands as a powerful tool, making system security more accessible and user-friendly.
Despite these challenges and limitations, the automation script serves as a valuable tool for organizations seeking to align with ANSSI guidelines and enhance the security posture of Linux systems.

May it serve as a valuable resource for users, educators, and security enthusiasts alike, contributing to the collective efforts in creating a safer digital environment.

7. References

https://cyber.gouv.fr/publications/configuration-recommendations-gnulinux-system
(PDF: Configuration recommendations of a gnu/linux system- v2)

https://blog.devgenius.io/automating-linux-system-hardening-scripted-hardening-for-debian-ubuntu-and-centos-4376396724ee
(Automating Linux System Hardening)