# Rocky.sh

```
echo
echo "======================================"
echo "    Rocky Linux 취약점 진단 시작    "
echo "======================================"
echo
```

```
OUTPUT_FILE="result.csv"
echo "항목,결과,설명" > "$OUTPUT_FILE"
exec >> "$OUTPUT_FILE" 2>&1
```

```
#1.1 root 계정 원격접속 제한 점검
SSH_CONFIG="/etc/ssh/sshd_config"

if [ ! -f "$SSH_CONFIG" ]; then
    echo "U-01,양호,SSH 서비스가 설치되어 있지 않거나 설정 파일이 존재하지 않
음"
    exit 0
fi

permit_root_login=$(grep -i "^PermitRootLogin" "$SSH_CONFIG" | awk '{pr
int $2}')

if [ "$permit_root_login" = "no" ]; then
    echo "U-01,양호,root 계정의 원격 접속이 제한됨 (PermitRootLogin no)"
else
    echo "U-01,취약,root 계정의 원격 접속이 허용됨 (PermitRootLogin 설정 필요)"
fi
```

```
#1.2 패스워드 복잡성 설정
PW_CONFIG="/etc/security/pwquality.conf"

if grep -Eiq "minlen\s*=\s*[8-9]|minlen\s*=\s*[1-9][0-9]+" "$PW_CONFIG"
```

```
&&
   grep -Eiq "dcredit\s*=\s*-\d+" "$PW_CONFIG" &&
   grep -Eiq "ucredit\s*=\s*-\d+" "$PW_CONFIG" &&
   grep -Eiq "lcredit\s*=\s*-\d+" "$PW_CONFIG" &&
   grep -Eiq "ocredit\s*=\s*-\d+" "$PW_CONFIG"; then
    echo "U-02,양호,패스워드 복잡성 설정이 적절히 구성됨"
else
    echo "U-02,취약,패스워드 복잡성 설정이 미흡함 (pwquality.conf 설정 필요)"
fi
```

```
#1.3 계정 잠금 임계값 설정
PAM_FILE="/etc/pam.d/system-auth"
REQUIRED_DENY=5
REQUIRED_UNLOCK_TIME=600

deny=$(grep -v '^[[:space:]]*#' "$PAM_FILE" | grep 'pam_faillock.so' | grep
-oE 'deny=[0-9]\+' | head -1 | cut -d= -f2)
unlock_time=$(grep -v '^[[:space:]]*#' "$PAM_FILE" | grep 'pam_faillock.s
o' | grep -oE 'unlock_time=[0-9]\+' | head -1 | cut -d= -f2)

if [ -z "$deny" ] || [ -z "$unlock_time" ]; then
  echo "U-03,취약,계정 잠금 임계값 설정 미흡"
else
  if [ "$deny" -ge "$REQUIRED_DENY" ] && [ "$unlock_time" -ge "$REQUIR
ED_UNLOCK_TIME" ]; then
    echo "U-03,양호,계정 잠금 임계값 적절히 설정됨"
  else
    echo "U-03,취약,계정 잠금 임계값 설정 미흡"
  fi
fi
```

```
#1.4 패스워드 파일 보호
PASSWD_FILE="/etc/passwd"
SHADOW_FILE="/etc/shadow"

if [[ ! -f $PASSWD_FILE || ! -f $SHADOW_FILE ]]; then
  echo "U-04,취약,패스워드 파일 존재하지 않음"
```

```
else
  passwd_perm=$(stat -c "%a" $PASSWD_FILE)
  shadow_perm=$(stat -c "%a" $SHADOW_FILE)
fi

if [[ "$passwd_perm" -le 644 && "$shadow_perm" -le 640 ]]; then
  echo "U-04,양호,패스워드 파일 권한 적절함"
else
  echo "U-04,취약,패스워드 파일 권한 부적절"
fi
```

```
#[1.5] 2.1 root홈, 패스 디렉터리 권한 및 패스 설정
PATH_VALUE="$PATH"
IS_VULNERABLE=0

case "$PATH_VALUE" in
  .:* | *:.:* )
    echo "U-05,취약,PATH 환경변수에 '.'이 앞이나 중간에 포함됨"
    IS_VULNERABLE=1
    ;;
esac

echo "$PATH_VALUE" | grep -q "::"
if [ $? -eq 0 ]; then
  echo "U-05,취약,PATH 환경변수에 빈 경로(::) 포함"
  IS_VULNERABLE=1
fi

if [ "$IS_VULNERABLE" -eq 0 ]; then
  echo "U-05,양호,PATH 환경변수에 '.' 또는 빈 경로(::)가 포함되지 않음"
fi
```

```
#[1.6] 2.2 파일 및 디렉터리 소유자 설정
CHECK_PATHS=("/etc" "/var" "/home" "/usr")

NOUSER_FILES=$(find "${CHECK_PATHS[@]}" -nouser 2>/dev/null)
NOGROUP_FILES=$(find "${CHECK_PATHS[@]}" -nogroup 2>/dev/null)
```

```
if [[ -z "$NOUSER_FILES" && -z "$NOGROUP_FILES" ]]; then
    echo "U-06,양호,소유자 및 그룹이 없는 파일이나 디렉터리가 없습니다."
else
    echo "U-06,취약,소유자 또는 그룹이 없는 파일/디렉터리가 발견되었습니다."
    if [[ -n "$NOUSER_FILES" ]]; then
        echo -e "\n소유자 없는 파일 및 디렉터리:"
        echo "$NOUSER_FILES"
    fi
    if [[ -n "$NOGROUP_FILES" ]]; then
        echo -e "\n그룹 없는 파일 및 디렉터리:"
        echo "$NOGROUP_FILES"
    fi
fi
```

```
#[1.7] 2.3 /etc/passwd 파일 소유자 및 권한 설정
FILE="/etc/passwd"

owner=$(stat -c "%U" "$FILE")
group=$(stat -c "%G" "$FILE")
perm=$(stat -c "%a" "$FILE")

if [[ "$owner" != "root" || "$group" != "root" || "$perm" -gt 644 ]]; then
  echo "U-07,취약,/etc/passwd 파일 소유자/그룹/권한 부적절"
else
  echo "U-07,양호,/etc/passwd 파일 소유자/그룹/권한 적절"
fi
```

```
#[1.8] 2.4 /etc/shadow 파일 소유자 및 권한 설정
FILE="/etc/shadow"
IS_VULNERABLE=0

if [ ! -e "$FILE" ]; then
  echo "U-08,취약,/etc/shadow 파일이 존재하지 않음"
  IS_VULNERABLE=1
else
  owner=$(stat -c "%U" "$FILE")
```

```
  group=$(stat -c "%G" "$FILE")
  perm=$(stat -c "%a" "$FILE")

  if [ "$owner" = "root" ] && [ "$group" = "root" ] && { [ "$perm" = "000" ] ||
| [ "$perm" = "400" ]; }; then
    echo "U-08,양호,/etc/shadow 파일 소유자 및 권한 설정 적절"
  else
    echo "U-08,취약,/etc/shadow 파일 소유자 및 권한 설정 미흡"
    IS_VULNERABLE=1
  fi
fi
```

```
#[1.9] 2.5 /etc/hosts 파일 소유자 및 권한 설정
FILE="/etc/hosts"
IS_VULNERABLE=0

if [ ! -e "$FILE" ]; then
  echo "U-09,취약,/etc/hosts 파일이 존재하지 않음"
  IS_VULNERABLE=1
else
  owner=$(stat -c "%U" "$FILE")
  group=$(stat -c "%G" "$FILE")
  perm=$(stat -c "%a" "$FILE")

  if [ "$owner" = "root" ] && [ "$group" = "root" ] && [ "$perm" -le 600 ]; th
en
    echo "U-09,양호,/etc/hosts 파일 소유자 및 권한 설정 적절"
  else
    echo "U-09,취약,/etc/hosts 파일 소유자 및 권한 설정 미흡"
    IS_VULNERABLE=1
  fi
fi
```

```
#[1.10] 2.6 /etc/(x)inetd.conf 파일 소유자 및 권한 설정
FILES=("/etc/inetd.conf" "/etc/xinetd.conf")
found=false
```

```
for FILE in "${FILES[@]}"; do
  if [ -e "$FILE" ]; then
    owner=$(stat -c "%U" "$FILE")
    group=$(stat -c "%G" "$FILE")
    perm=$(stat -c "%a" "$FILE")

    if [[ "$owner" == "root" && "$group" == "root" && "$perm" -le 600 ]]; th
en
      echo "U-10,양호,$FILE 소유자 및 권한 설정 적절"
    else
      echo "U-10,취약,$FILE 소유자 및 권한 설정 미흡"
    fi
    found=true
  fi
done

if ! $found; then
  echo "U-10,양호,/etc/(x)inetd.conf 파일 존재하지 않음"
fi
```

```
#[1.11] 2.7 /etc/syslog.conf 파일 소유자 및 권한 설정
FILES=("/etc/syslog.conf")
found=false

for FILE in "${FILES[@]}"; do
  if [ -e "$FILE" ]; then
    owner=$(stat -c "%U" "$FILE")
    group=$(stat -c "%G" "$FILE")
    perm=$(stat -c "%a" "$FILE")

    if [[ "$owner" == "root" && "$group" == "root" && "$perm" -le 640 ]]; th
en
      echo "U-11,양호,$FILE 소유자 및 권한 설정 적절"
    else
      echo "U-11,취약,$FILE 소유자 및 권한 설정 미흡"
    fi
    found=true
  fi
```

```
done

if ! $found; then
  echo "U-11,양호,/etc/syslog.conf  파일 존재하지 않음"
fi
```

```
#[1.12] 2.8 /etc/services 파일 소유자 및 권한 설정
FILE="/etc/services"
IS_VULNERABLE=0

if [ ! -e "$FILE" ]; then
  echo "U-12,취약,/etc/services 파일이 존재하지 않음"
  IS_VULNERABLE=1
fi

owner=$(stat -c "%U" "$FILE")
group=$(stat -c "%G" "$FILE")
perm=$(stat -c "%a" "$FILE")

if [[ "$owner" == "root" && "$group" == "root" && "$perm" -le 644 ]]; then
  echo "U-12,양호,/etc/services 파일 소유자 및 권한 설정 적절"
else
  echo "U-12,취약,/etc/services 파일 소유자 및 권한 설정 미흡"
fi
```

```
#[1.13] 2.9 SUID, SGID, 설정 파일점검
SUID_FILES=$(find / -type f -perm -4000 2>/dev/null)
SGID_FILES=$(find / -type f -perm -2000 2>/dev/null)

if [[ -z "$SUID_FILES" && -z "$SGID_FILES" ]]; then
    echo "U-13,양호,SUID, SGID 설정된 파일이 존재하지 않음"
else
    echo "U-13,취약,SUID, SGID 설정된 파일 존재"
fi
```

```
#[1.14] 2.10 사용자, 시스템 시작파일 및 환경파일 소유자 및 권한 설정
users=$(awk -F: '$3 >= 1000 && $1 != "nobody" {print $1}' /etc/passwd)
files_to_check=(".profile" ".bash_profile" ".bashrc" ".cshrc" ".kshrc")

vulnerable_files=()

for user in $users; do
  home_dir=$(eval echo ~$user)

  for file in "${files_to_check[@]}"; do
    target_file="$home_dir/$file"
    if [ -f "$target_file" ]; then
      owner=$(stat -c "%U" "$target_file")
      perm=$(stat -c "%a" "$target_file")
      if [[ "$owner" != "$user" && "$owner" != "root" ]] || (( perm & 22 )); the
n
        vulnerable_files+=("$target_file")
      fi
    fi
  done
done

if [ ${#vulnerable_files[@]} -eq 0 ]; then
  echo "U-14,양호,홈 디렉터리 환경파일의 소유자 및 권한 설정 적절"
else
  files_str=$(printf "%s\n" "${vulnerable_files[@]}")
  echo "U-14,취약,홈 디렉터리 환경파일의 소유자 및 권한 설정 미흡($files_str)"
fi
```

```
#[1.15] 2.11 world writable 파일 점검
WORLD_WRITABLE_FILES=$(find / -xdev -type f -perm -0002 2>/dev/null)

if [ -z "$WORLD_WRITABLE_FILES" ]; then
  echo "U-15,양호,world writable 파일이 존재하지 않음"
else
```

```
  echo "U-15,취약,world writable 파일 존재($WORLD_WRITABLE_FILES)"
fi
```

```
#[1.16] 2.12 /dev에 존재하지 않는 device 파일 점검
found_invalid=()

for file in /dev/*; do
  if [ ! -e "$file" ]; then
    continue
  fi
  if [ -b "$file" ] || [ -c "$file" ]; then
    dev_major=$(stat -c "%t" "$file")
    dev_minor=$(stat -c "%T" "$file")
    if ! ls -l /dev | grep -q "$(basename "$file")"; then
      found_invalid+=("$file")
    fi
  else
    if [ ! -e "$file" ]; then
      found_invalid+=("$file")
    fi
  fi
done

if [ ${#found_invalid[@]} -eq 0 ]; then
  echo "U-16,양호,/dev에 존재하지 않는 device 파일이 없음"
else
  echo "U-16,취약,/dev에 존재하지 않는 device 파일 존재(${found_invalid[*]})"
fi
```

```
#[1.17] 2.13 $HOME/.rhosts, hosts.equiv 사용 금지
bad_files=()

check_file() {
  file=$1
  expected_owner=$2

  if [ -f "$file" ]; then
```

```
    perm=$(stat -c "%a" "$file")
    owner=$(stat -c "%U" "$file")

    if [ "$owner" != "$expected_owner" ] && [ "$owner" != "root" ]; then
      bad_files+=("$file(소유자 오류)")
      return
    fi

    if [ "$perm" -gt 600 ]; then
      bad_files+=("$file(권한 오류: $perm)")
      return
    fi

    if grep -q "^\+" "$file"; then
      bad_files+=("$file(+ 설정 포함)")
      return
    fi
  fi
}

check_file /etc/hosts.equiv root
check_file /etc/rhosts root

while IFS=: read -r user _ uid _ home _; do
  if [ "$uid" -ge 1000 ] && [ -d "$home" ]; then
    user_rhosts="$home/.rhosts"
    check_file "$user_rhosts" "$user"
  fi
done < /etc/passwd

if [ ${#bad_files[@]} -eq 0 ]; then
  echo "U-17,양호,.rhosts, hosts.equiv, rhosts 파일이 없거나 보안 설정이 적절함"
else
  echo "U-17,취약,다음 파일이 보안 기준에 부적합함:"
  for f in "${bad_files[@]}"; do
    echo " - $f"
  done
    fi
```

```
#[U-18] 2.14 접속 IP 및 포트 제한
issue_found=false

if command -v iptables > /dev/null 2>&1; then
  if iptables -L INPUT -n | grep -E 'ACCEPT|DROP' | grep -q '0.0.0.0/0'; then
    issue_found=true
  fi
fi

if [ -f /etc/hosts.allow ]; then
  if ! grep -qvE '^\s*#|^$' /etc/hosts.allow; then
    issue_found=true
  fi
fi

if [ -f /etc/hosts.deny ]; then
  if ! grep -q "ALL: ALL" /etc/hosts.deny; then
    issue_found=true
  fi
fi

if [ "$issue_found" = false ]; then
  echo "U-18,양호,IP 및 포트 접근이 제한되어 있음"
else
  echo "U-18,취약,IP 또는 포트 제한 설정이 부적절함"
fi
```

```
#[U-19] 3.1 Finger 서비스 비활성화
if systemctl is-enabled finger.service >/dev/null 2>&1; then
  echo "U-19,취약,Finger 서비스가 활성화 되어 있음"
else
  echo "U-19,양호,Finger 서비스가 비활성화 되어 있음"
fi
```

```
#[U-20] 3.2 Anonymous FTP 비활성화
if systemctl is-active vsftpd.service >/dev/null 2>&1; then
```

```
 if grep -q '^anonymous_enable=NO' /etc/vsftpd/vsftpd.conf 2>/dev/null; t
hen
   echo "U-20,양호,익명 FTP 접속이 차단되어 있음"
 else
   echo "U-20,취약,익명 FTP 접속이 허용되어 있음"
 fi
else
 echo "U-20,양호,vsftpd 서비스 비활성화"
fi
```

```
# [U-21] 3.3 r 계열 서비스 비활성화
services=("rlogin.service" "rsh.service" "rexec.service")

issue_found=false

for svc in "${services[@]}"; do
 if systemctl is-enabled "$svc" >/dev/null 2>&1; then
   issue_found=true
   break
 fi
done

if [ "$issue_found" = false ]; then
 echo "U-21,양호,불필요한 r 계열 서비스가 비활성화 되어 있음"
else
 echo "U-21,취약,불필요한 r 계열 서비스가 활성화 되어 있음"
fi
```

```
#[U-22] 3.4 crond 파일 소유자 및 권한 설정
issue_found=false

if [ -x "$(command -v crontab)" ]; then
 [[ $(stat -c "%a" "$(command -v crontab)") -gt 750 ]] && issue_found=tru
e
else
 issue_found=true
fi
```

```
for f in /etc/crontab /etc/cron.allow /etc/cron.d /var/spool/cron; do
  [[ -e $f && $(stat -c "%a" "$f") -gt 640 ]] && issue_found=true && break
done

if ! $issue_found; then
  echo "U-22,양호,crontab 명령어 일반사용자 금지 및 cron 관련 파일 권한 적정"
else
  echo "U-22,취약,crontab 권한 또는 cron 파일 권한 부적절"
fi
```

```
#[U-23] 3.5 DoS 공격에 취약한 서비스 비활성화
services=(
  chargen-dgram@udp.service
  chargen-stream@tcp.service
  daytime-dgram@udp.service
  daytime-stream@tcp.service
  echo-dgram@udp.service
  echo-stream@tcp.service
  tcpmux-server.service
)

issue_found=false

for svc in "${services[@]}"; do
  if systemctl is-enabled "$svc" &>/dev/null; then
    issue_found=true
    break
  fi
done

if ! $issue_found; then
  echo "U-23,양호,DoS 공격에 취약한 서비스가 비활성화 되어 있음"
else
  echo "U-23,취약,DoS 공격에 취약한 서비스가 활성화 되어 있음"
fi
```

```
#[U-24] 3.6 NFS 서비스 비활성화
services=(
  nfs-server.service
  rpcbind.service
)

issue_found=false

for svc in "${services[@]}"; do
  if systemctl is-enabled "$svc" &>/dev/null; then
    issue_found=true
    break
  fi
done

if ! $issue_found; then
  echo "U-24,양호,불필요한 NFS 서비스 관련 데몬이 비활성화 되어 있음"
else
  echo "U-24,취약,불필요한 NFS 서비스 관련 데몬이 활성화 되어 있음"
fi
```

```
#[U-25] 3.7 NFS 접근 통제
exports_file="/etc/exports"
IS_VULNERABLE=0

if [ ! -f "$exports_file" ]; then
  echo "U-25,양호,/etc/exports 파일이 존재하지 않음 (NFS 사용 안 함)"
else
  if grep -q '^[^#]*\s\+\*\( \|$\)' "$exports_file"; then
    echo "U-25,취약,/etc/exports 파일에 everyone(*) 공유 설정이 있음"
    IS_VULNERABLE=1
  else
    echo "U-25,양호,/etc/exports 파일에 everyone(*) 공유 설정이 없음"
  fi
fi
```

```
#[U-26] 3.8 automountd 제거
if systemctl list-unit-files | grep -q autofs; then
  state=$(systemctl is-enabled autofs 2>/dev/null)
  if [ "$state" = "enabled" ]; then
    echo "U-26,취약,autofs(automountd) 서비스가 활성화되어 있음"
  else
    echo "U-26,양호,autofs(automountd) 서비스가 비활성화되어 있음"
  fi
else
  echo "U-26,양호,autofs(automountd) 서비스가 설치되어 있지 않음"
fi
```

```
#[U-27] 3.9 RPC 서비스 확인
services=("rpcbind" "rpc-statd" "rpc-idmapd")
vulnerable=()

for svc in "${services[@]}"; do
  if systemctl list-unit-files | grep -q "$svc"; then
    state=$(systemctl is-enabled "$svc" 2>/dev/null)
    if [ "$state" = "enabled" ]; then
      vulnerable+=("$svc")
    fi
  fi
done

if [ ${#vulnerable[@]} -eq 0 ]; then
  echo "U-27,양호,불필요한 RPC 서비스가 비활성화되어 있음"
else
  echo "U-27,취약,다음 RPC 서비스가 활성화되어 있음 (${vulnerable[*]})"
fi
```

```
#[U-28] 3.10 NIS, NIS+ 점검
services=("ypserv" "ypbind" "yppasswdd" "rpc.yppasswdd")
vulnerable=()

for svc in "${services[@]}"; do
```

```
  if systemctl list-unit-files | grep -q "$svc"; then
    state=$(systemctl is-enabled "$svc" 2>/dev/null)
    if [ "$state" = "enabled" ]; then
      vulnerable+=("$svc")
    fi
  fi
done

if [ ${#vulnerable[@]} -eq 0 ]; then
  echo "U-28,양호,NIS 서비스가 비활성화되어 있음"
else
  echo "U-28,취약,다음 NIS 서비스가 활성화되어 있음 (${vulnerable[*]})"
fi
```

```
#[U-29] 3.11 tftp, talk 서비스 비활성화
services=("tftp" "talk" "ntalk")
vulnerable=()

for svc in "${services[@]}"; do
  if systemctl list-unit-files | grep -q "$svc"; then
    state=$(systemctl is-enabled "$svc" 2>/dev/null)
    if [ "$state" = "enabled" ]; then
      vulnerable+=("$svc")
    fi
  fi
done

if [ ${#vulnerable[@]} -eq 0 ]; then
  echo "U-29,양호,tftp, talk, ntalk 서비스가 비활성화되어 있음"
else
  echo "U-29,취약,다음 서비스가 활성화되어 있음 (${vulnerable[*]})"
fi
```

```
#[U-30] Sendmail 버전 점검
current_version=$(sendmail -d0.1 -bv root 2>/dev/null | grep 'Version' | aw
k '{print $2}')
recommended_version="8.18.1"
```

```
if ! command -v sendmail &> /dev/null; then
  echo "U-30,양호,Sendmail이 설치되어 있지 않음"
elif [ "$current_version" = "$recommended_version" ]; then
  echo "U-30,양호,Sendmail이 최신 버전($recommended_version)"
else
  echo "U-30,취약,Sendmail 버전이 최신이 아님 (현재: $current_version, 권장:
$recommended_version)"
fi
```

```
#[U-31] 3.13 스팸 메일 릴레이 제한
services=("sendmail" "postfix")
vulnerable=()

check_sendmail() {
  grep -i "PrivacyOptions" /etc/mail/sendmail.mc 2>/dev/null | grep -q "noe
xpn,nouucp,restrictqrun,restrictmailq,authwarnings"
}

check_postfix() {
  grep -i "smtpd_recipient_restrictions" /etc/postfix/main.cf 2>/dev/null | gr
ep -Eiq "reject_unauth_destination"
}

for svc in "${services[@]}"; do
  if systemctl list-unit-files | grep -q "$svc"; then
    if [ "$(systemctl is-enabled "$svc" 2>/dev/null)" = "enabled" ]; then
      if [ "$svc" = "sendmail" ]; then
        if ! check_sendmail; then
          vulnerable+=("$svc")
        fi
      elif [ "$svc" = "postfix" ]; then
        if ! check_postfix; then
          vulnerable+=("$svc")
        fi
      fi
    fi
  fi
```

```
done

if [ ${#vulnerable[@]} -eq 0 ]; then
  echo "U-31,양호,SMTP 서비스가 없거나 릴레이 제한이 설정되어 있음"
else
  echo "U-31,취약,다음 SMTP 서비스에서 릴레이 제한이 설정되어 있지 않음 (${vul
nerable[*]})"
fi
```

```
#[U-32] 3.14 일반사용자의 Sendmail 실행 방지
service="sendmail"

if ! systemctl list-unit-files | grep -q "$service"; then
  echo "U-32,양호,Sendmail이 설치되어 있지 않음 또는 사용하지 않음"
elif grep -q "^O RestrictMailq=restrictqrun" /etc/mail/sendmail.cf 2>/dev/nu
ll; then
  echo "U-32,양호,일반 사용자의 Sendmail 실행이 제한되어 있음"
else
  echo "U-32,취약,일반 사용자의 Sendmail 실행 제한이 설정되어 있지 않음"
fi
```

```
#[U-33] 3.15 DNS 보안 버전 패치
service="named"
recommended_version="9.18.24"

if ! command -v named &> /dev/null; then
  echo "U-33,양호,DNS 서비스를 사용하지 않음"
else
  current_version=$(named -v | awk '{print $2}')
  if [ "$current_version" = "$recommended_version" ]; then
    echo "U-33,양호,DNS 서비스가 최신 버전($recommended_version)"
  else
    echo "U-33,취약,DNS 서비스 버전이 최신이 아님 (현재: $current_version, 권
장: $recommended_version)"
  fi
fi
```

```
#[U-34] 3.16 DNS Zone Transfer 설정
config_file="/etc/named.conf"

if ! command -v named &> /dev/null; then
  echo "U-34,양호,DNS 서비스를 사용하지 않음"
elif [ ! -f "$config_file" ]; then
  echo "U-34,취약,named.conf 파일이 존재하지 않음"
elif grep -q "allow-transfer" "$config_file"; then
  if grep "allow-transfer" "$config_file" | grep -qv "any"; then
    echo "U-34,양호,Zone Transfer가 허가된 사용자에게만 허용됨"
  else
    echo "U-34,취약,Zone Transfer가 모든 사용자(any)에게 허용됨"
  fi
else
  echo "U-34,취약,Zone Transfer 설정이 없음 (기본값은 모든 사용자에게 허용)"
fi
```

```
#[U-35] 3.17 웹서비스 디렉토리 리스팅 제거
httpd_conf="/etc/conf/httpd.conf"

if ! command -v httpd &> /dev/null; then
  echo "U-35,양호,웹서비스(Apache)를 사용하지 않음"
elif [ ! -f "$httpd_conf" ]; then
  echo "U-35,취약,Apache 설정 파일이 존재하지 않음"
elif grep -q "Options Indexes" "$httpd_conf"; then
  echo "U-35,취약,디렉토리 리스팅(Indexes)이 허용되어 있음"
else
  echo "U-35,양호,디렉토리 리스팅이 제거되어 있음"
fi
```

```
#[U-36] 3.18 웹서비스 웹 프로세스 권한 제한
web_processes=("httpd" "apache2")
running_user=$(ps -eo user:20,comm | grep -E "${web_processes[*]}" | gr
ep -v root | awk '{print $1}' | sort | uniq)

if ! pgrep -x httpd > /dev/null && ! pgrep -x apache2 > /dev/null; then
```

```
  echo "U-36,양호,웹서비스가 구동되고 있지 않음"
elif echo "$running_user" | grep -qw "root"; then
  echo "U-36,취약,웹 프로세스가 root 권한으로 실행 중"
elif [ -n "$running_user" ]; then
  echo "U-36,양호,웹 프로세스가 일반 사용자($running_user) 권한으로 실행 중"
else
  echo "U-36,취약,웹 프로세스 실행 사용자 확인 불가"
fi
```

```
#[U-37] 3.19 웹서비스 상위 디렉토리 접근 금지
httpd_conf="/etc/httpd/conf/httpd.conf"
access_conf="/etc/httpd/conf.d/access.conf"

file_to_check=""

if [ -f "$httpd_conf" ]; then
  file_to_check="$httpd_conf"
elif [ -f "$access_conf" ]; then
  file_to_check="$access_conf"
fi

if [ -z "$file_to_check" ]; then
  echo "U-37,양호,Apache 설정 파일이 존재하지 않음 (웹 서비스 미사용)"
elif grep -i "<Directory" "$file_to_check" | grep -q '\.\.'; then
  echo "U-37,취약,상위 디렉토리 접근 허용 설정이 존재함"
else
  echo "U-37,양호,상위 디렉토리 접근 제한 설정이 적용됨"
fi
```

```
#[U-38] 3.20 웹서비스 불필요한 파일 제거
apache_paths=("/var/www/html/manual" "/var/www/manual" "/etc/httpd/ht
docs/manual" "/etc/httpd/manual")
vulnerable=()

for path in "${apache_paths[@]}"; do
  if [ -e "$path" ]; then
    vulnerable+=("$path")
```

```
  fi
done

if [ ${#vulnerable[@]} -eq 0 ]; then
  echo "U-38,양호,불필요한 Apache 매뉴얼 디렉터리가 존재하지 않음"
else
  echo "U-38,취약,다음 불필요한 디렉터리가 존재함 (${vulnerable[*]})"
fi
```

```
#[U-39] 3.21 웹서비스 링크 사용금지
apache_conf="/etc/httpd/conf/httpd.conf"
vulnerable=()

if grep -E 'Options\s+.*FollowSymLinks' "$apache_conf" >/dev/null 2>&1; t
hen
  vulnerable+=("FollowSymLinks 옵션")
fi

if grep -E '^\s*Alias\s+' "$apache_conf" >/dev/null 2>&1; then
  vulnerable+=("Alias 설정")
fi

if [ ${#vulnerable[@]} -eq 0 ]; then
  echo "U-39,양호,심볼릭 링크(FollowSymLinks) 및 Alias 사용이 제한됨"
else
  echo "U-39,취약,다음 항목이 설정되어 있음 (${vulnerable[*]})"
fi
```

```
#[U-40] 3.22 웹서비스 파일 업로드 및 다운로드 제한
apache_conf="/etc/httpd/conf/httpd.conf"
limit_value=$(grep -E '^\s*LimitRequestBody\s+[0-9]+' "$apache_conf" | a
wk '{print $2}')

if [ -z "$limit_value" ]; then
  echo "U-40,취약,LimitRequestBody 설정이 없음"
else
  if [ "$limit_value" -le 5000000 ]; then
```

```
    echo "U-40,양호,LimitRequestBody가 $limit_value 바이트로 제한됨"
  else
    echo "U-40,취약,LimitRequestBody가 $limit_value 바이트로 너무 크게 설정
됨"
  fi
fi
```

```
#[U-41] 3.23 웹서비스 영역의 분리
apache_conf="/etc/httpd/conf/httpd.conf"
docroot=$(grep -E '^\s*DocumentRoot\s+"' "$apache_conf" | awk '{print
$2}' | tr -d '"')

default_paths=(
  "/usr/local/apache/htdocs"
  "/usr/local/apache2/htdocs"
  "/var/www/html"
)

if [ -z "$docroot" ]; then
  echo "U-41,취약,DocumentRoot 설정이 없음"
else
  if [[ " ${default_paths[*]} " == *" $docroot "* ]]; then
    echo "U-41,양호,DocumentRoot가 별도 디렉터리로 지정됨 ($docroot)"
  else
    echo "U-41,취약,DocumentRoot가 기본 경로가 아님 ($docroot)"
  fi
fi
```

```
#[U-42] 4.1 최신 보안패치 및 벤더 권고사항 적용
policy_file="/etc/security/patch_policy.conf"
patch_log="/var/log/patch_management.log"

if [ -f "$policy_file" ]; then
  policy_status="수립됨"
else
  policy_status="수립되지 않음"
fi
```

```
if [ -s "$patch_log" ]; then
  patch_status="주기적으로 패치 적용 확인됨"
else
  patch_status="패치 적용 내역 확인 불가 또는 없음"
fi

if [[ "$policy_status" == "수립됨" && "$patch_status" == "주기적으로 패치 적용 확인됨" ]]; then
  echo "U-42,양호,패치 정책이 수립되어 있으며, 정책에 따른 패치 적용 내역이 존재함"
else
  echo "U-42,취약,패치 정책 수립 또는 적용 내역이 부적절함 (정책: $policy_status, 적용: $patch_status)"
fi
```

```
#[U-43] 5.1 로그의 정기적 검토 및 보고
vulnerable=()

if [ -e /var/log/wtmp ]; then
  last_login=$(last -n 1 | head -1)
else
  last_login="No wtmp log found"
  vulnerable+=("wtmp 로그 없음")
fi

if [ -e /var/log/btmp ]; then
  failed_login=$(lastb -n 1 | head -1)
else
  failed_login="No btmp log found"
  vulnerable+=("btmp 로그 없음")
fi

if [ -e /var/log/sulog ]; then
  su_attempts=$(tail -20 /var/log/sulog)
else
  su_attempts="No sulog found"
  vulnerable+=("sulog 로그 없음")
```

```
fi

if [ -e /var/log/xferlog ]; then
  ftp_access=$(tail -20 /var/log/xferlog)
else
  ftp_access="No xferlog found"
  vulnerable+=("xferlog 로그 없음")
fi

if [ ${#vulnerable[@]} -eq 0 ]; then
  echo "U-43,양호,로그 파일들이 정상적으로 존재하며 검토 가능"
else
  echo "U-43,취약,누락된 로그 파일 - ${vulnerable[*]}"
fi
```

```
#[U-44] 1.5 root 이외의 UID가 '0' 금지
vulnerable=()

while IFS=: read -r username _ uid _; do
  if [ "$uid" = "0" ] && [ "$username" != "root" ]; then
    vulnerable+=("$username")
  fi
done < /etc/passwd

if [ ${#vulnerable[@]} -eq 0 ]; then
  echo "U-44,양호,root 외에 UID 0을 가진 계정이 없음"
else
  echo "U-44,취약,root 외에 UID 0을 가진 계정 존재 (${vulnerable[*]})"
fi
```

```
#[U-45] 1.6 root 계정 su 제한
pam_file="/etc/pam.d/su"
group_name="wheel"

if grep -Eq "^auth\s+required\s+pam_wheel.so" "$pam_file"; then
  echo "U-45,양호,su 명령어 사용이 '$group_name' 그룹으로 제한되어 있음"
else
```

```
  echo "U-45,취약,su 명령어 사용 제한 설정이 없음"
fi
```

```
#[U-46] 1.7 패스워드 최소 길이 설정
login_defs="/etc/login.defs"
minlen=$(grep -E '^PASS_MIN_LEN' "$login_defs" | awk '{print $2}')

if [ -z "$minlen" ]; then
  echo "U-46,취약,패스워드 최소 길이 설정이 존재하지 않음"
elif [ "$minlen" -ge 8 ]; then
  echo "U-46,양호,패스워드 최소 길이가 8자 이상으로 설정되어 있음"
else
  echo "U-46,취약,패스워드 최소 길이가 8자 미만으로 설정되어 있음 (현재: $minle
n)"
fi
```

```
#[U-47] 1.8 패스워드 최대 사용기간 설정
login_defs="/etc/login.defs"
maxdays=$(grep -E '^PASS_MAX_DAYS' "$login_defs" | awk '{print $2}')

if [ -z "$maxdays" ]; then
  echo "U-47,취약,패스워드 최대 사용기간 설정이 존재하지 않음"
elif [ "$maxdays" -le 90 ]; then
  echo "U-47,양호,패스워드 최대 사용기간이 90일 이하로 설정되어 있음 (현재: $m
axdays)"
else
  echo "U-47,취약,패스워드 최대 사용기간이 90일 초과로 설정되어 있음 (현재: $m
axdays)"
fi
```

```
#[U-48] 1.9 패스워드 최소 사용기간 설정
users=$(awk -F: '{ if ($3 >= 1000 && $1 != "nobody") print $1 }' /etc/passw
d)
vulnerable_users=()

for user in $users; do
```

```
  min_days=$(chage -l "$user" | grep "Minimum number of days between p
assword change" | awk -F: '{print $2}' | tr -d ' ')

  if [ -z "$min_days" ]; then
    vulnerable_users+=("$user (설정 확인 불가)")
  elif [ "$min_days" -lt 1 ]; then
    vulnerable_users+=("$user ($min_days 일)")
  fi
done

if [ ${#vulnerable_users[@]} -eq 0 ]; then
  echo "U-48,양호,모든 일반 사용자의 패스워드 최소 사용기간이 1일 이상으로 설정
됨"
else
  echo "U-48,취약,다음 사용자들의 최소 사용기간이 기준 미만임"
  for user in "${vulnerable_users[@]}"; do
    echo " - $user"
  done
fi
```

```
#[U-49] 1.10 불필요한 계정 제거
unused_accounts=()
suspicious_accounts=()

default_accounts=("lp" "uucp" "nuucp" "sync" "shutdown" "halt" "news"
"operator" "games" "gopher")

for account in "${default_accounts[@]}"; do
  if grep -q "^${account}:" /etc/passwd; then
    suspicious_accounts+=("$account")
  fi
done

last_login_check() {
  local user=$1
  lastlog -u "$user" | grep -v "Never logged in" | grep -q .
}
```

```
for user in $(cut -d: -f1 /etc/passwd); do
  if ! last_login_check "$user"; then
    unused_accounts+=("$user")
  fi
done

if [ ${#suspicious_accounts[@]} -eq 0 ] && [ ${#unused_accounts[@]} -eq
0 ]; then
  echo "U-49,양호,불필요한 계정이 존재하지 않음"
else
  echo "U-49,취약,불필요한 계정 또는 미사용 계정 존재"
fi
```

```
#[U-50] 1.11 관리자 그룹에 최소한의 계정 포함
admin_group="root"
group_file="/etc/group"
IS_VULNERABLE=0

if ! grep -q "^${admin_group}:" "$group_file"; then
  echo "U-50,취약,관리자 그룹 '${admin_group}'이 존재하지 않음"
  IS_VULNERABLE=1
else
  accounts=$(grep "^${admin_group}:" "$group_file" | cut -d: -f4)
  if [ -z "$accounts" ]; then
    echo "U-50,취약,관리자 그룹 '${admin_group}'에 등록된 계정이 없음"
    IS_VULNERABLE=1
  else
    echo "U-50,양호,관리자 그룹 '${admin_group}'에 등록된 계정이 있음 ($acco
unts)"
  fi
fi
```

```
#[U-51] 1.12 계정이 존재하지 않는 GID 금지
group_file="/etc/group"
passwd_file="/etc/passwd"

gid_list=$(cut -d: -f3 "$group_file" | sort -u)
```

```
existing_gids=$(cut -d: -f4 "$passwd_file" | sort -u)

invalid_gids=()

for gid in $gid_list; do
  if ! grep -q "^[^:]*:[^:]*:$gid:" "$passwd_file"; then
    invalid_gids+=("$gid")
  fi
done

if [ ${#invalid_gids[@]} -eq 0 ]; then
  echo "U-51,양호,계정이 존재하지 않는 GID가 없음"
else
  echo "U-51,취약,계정이 존재하지 않는 GID가 존재함 (${invalid_gids[*]})"
fi
```

```
#[U-52] 1.13 동일한 UID 금지
declare -A uid_map
duplicates=()

while IFS=: read -r username _ uid _ _ _ _; do
  if [[ -n "$uid" ]]; then
    if [[ -n "${uid_map[$uid]}" ]]; then
      duplicates+=("$uid (${uid_map[$uid]}, $username)")
    else
      uid_map[$uid]=$username
    fi
  fi
done < /etc/passwd

if [ ${#duplicates[@]} -eq 0 ]; then
  echo "U-52,양호,동일한 UID가 존재하지 않습니다."
else
  echo "U-52,취약,동일한 UID가 중복된 계정이 있습니다. 중복 UID 목록:"
  for entry in "${duplicates[@]}"; do
    echo " - $entry"
```

```
  done
fi
```

```
#[U-53] 1.14 사용자 shell 점검
nologin_users=(bin daemon adm lp sync shutdown halt mail operator game
s ftp nobody)
misconfigured=()

for user in "${nologin_users[@]}"; do
  shell=$(getent passwd "$user" | cut -d: -f7)
  if [[ "$shell" != "/sbin/nologin" && "$shell" != "/bin/false" ]]; then
    misconfigured+=("$user ($shell)")
  fi
done

if [ ${#misconfigured[@]} -eq 0 ]; then
  echo "U-53,양호,로그인 불필요 계정에 적절한 쉘이 설정되어 있습니다."
else
  echo "U-53,취약,다음 계정에 적절하지 않은 쉘이 설정되어 있습니다:"
  for entry in "${misconfigured[@]}"; do
    echo " - $entry"
  done
fi
```

```
#[U-54] 1.15 Session Timeout 설정
files=("/etc/profile" "/etc/bashrc")
files+=(/etc/profile.d/*.sh)

is_secure=true
details=()

for file in "${files[@]}"; do
  if [ -f "$file" ]; then
    tmout_line=$(grep -E '^\s*TMOUT\s*=' "$file" 2>/dev/null | tail -n 1)
    if [[ -n "$tmout_line" ]]; then
      value=$(echo "$tmout_line" | cut -d= -f2 | tr -d ' ')
      if [[ "$value" =~ ^[0-9]+$ ]]; then
```

```
        if [ "$value" -gt 600 ]; then
          is_secure=false
          details+=("$file: TMOUT=$value (600초 초과)")
        fi
      else
        is_secure=false
        details+=("$file: TMOUT 값이 숫자가 아님")
      fi
    else
      is_secure=false
      details+=("$file: TMOUT 설정 없음")
    fi
  fi
done

if $is_secure; then
  echo "U-54,양호,Session Timeout이 적절하게 설정되어 있습니다."
else
  echo "U-54,취약,다음 파일에 설정이 누락되었거나 600초 초과 설정이 존재합니
다."
    for line in "${details[@]}"; do
    echo " - $line"
  done
fi
```

```
#[U-55] 2.15 hosts.lpd 파일 소유자 및 권한 설정
file="/etc/hosts.lpd"

if [ ! -e "$file" ]; then
  echo "U-55,양호,hosts.lpd 파일이 존재하지 않습니다."
else
  owner=$(stat -c "%U" "$file")
  perm=$(stat -c "%a" "$file")

  if [ "$owner" == "root" ] && [ "$perm" == "600" ]; then
    echo "U-55,양호,hosts.lpd 파일의 소유자 및 권한이 적절합니다."
  else
    echo "U-55,취약,hosts.lpd 파일의 보안 설정이 적절하지 않습니다."
```

```
    echo " - 현재 소유자: $owner"
    echo " - 현재 권한: $perm"
  fi
fi
```

```
#[U-56] 2.17 UMASK 설정 관리
files=("/etc/profile" "/etc/bashrc" "/etc/login.defs" "/etc/csh.cshrc")
found=0
vulnerable=()

for file in "${files[@]}"; do
  if [ -f "$file" ]; then
    umask_lines=$(grep -E '^[^#]*umask[[:space:]]+[0-9]+' "$file")
    if [ -n "$umask_lines" ]; then
      while read -r line; do
        umask_val=$(echo "$line" | grep -oE '[0-7]{3}')
        if [ "$umask_val" -lt 022 ]; then
          vulnerable+=("$file: $line")
        else
          found=1
        fi
      done <<< "$umask_lines"
    fi
  fi
done

if [ ${#vulnerable[@]} -gt 0 ]; then
  echo "U-56,취약,다음 파일에서 umask 설정이 022 미만입니다:"
  for v in "${vulnerable[@]}"; do
    echo " - $v"
  done
elif [ $found -eq 1 ]; then
  echo "U-56,양호,모든 umask 설정이 022 이상입니다."
else
  echo "U-56,취약,명시적인 UMASK 설정이 없습니다."
fi
```

```
#[U-57] 2.18 홈디렉토리 소유자 및 권한 설정
vulnerable=()

while IFS=: read -r username _ uid _ _ homedir _; do
  if [ "$uid" -ge 1000 ] && [ -d "$homedir" ]; then
    owner=$(stat -c %U "$homedir")
    perm=$(stat -c %a "$homedir")
    other_write=$((perm % 10))

    if [[ "$owner" != "$username" || "$other_write" -ge 2 ]]; then
      vulnerable+=("$username ($homedir - owner: $owner, perm: $perm)")
    fi
  fi
done < /etc/passwd

if [ ${#vulnerable[@]} -eq 0 ]; then
  echo "U-57,양호,모든 홈 디렉토리 소유자와 권한이 적절합니다."
else
  echo "U-57,취약,다음 계정의 홈 디렉토리 설정이 잘못되어 있습니다:"
  for entry in "${vulnerable[@]}"; do
    echo " - $entry"
  done
f
```

```
#[U-58] 2.19 홈디렉토리로 지정한 디렉토리의 존재 관리
vuln_users=""

while IFS=: read username _ uid _ _ home shell
do
  if [ "$uid" -ge 1000 ] && [ "$username" != "nobody" ]; then
    if [ ! -d "$home" ]; then
      vuln_users="$vuln_users $username($home)"
    fi
  fi
done < /etc/passwd
```

```
if [ -z "$vuln_users" ]; then
  echo "U-58,양호,모든 계정의 홈 디렉터리가 존재합니다."
else
  echo "U-58,취약,다음 계정의 홈 디렉터리가 존재하지 않습니다:"
  for user in $vuln_users; do
    echo " - $user"
  done
fi
```

```
#[U-59] 2.20 숨겨진 파일 및 디렉토리 검색 및 제거
search_dir="/"

hidden_files=$(find "$search_dir" -type f -name ".*" 2>/dev/null)
hidden_dirs=$(find "$search_dir" -type d -name ".*" 2>/dev/null)

if [ -z "$hidden_files" ] && [ -z "$hidden_dirs" ]; then
  echo "U-59,양호,숨겨진 파일 및 디렉터리가 존재하지 않거나 모두 정리됨"
else
  echo "U-59,취약,다음 숨겨진 파일 및 디렉터리가 존재합니다."
fi
```

```
#[U-60] 3.24 ssh 원격접속 허용
ssh_status=$(systemctl is-active sshd 2>/dev/null)
telnet_installed=$(rpm -q telnet-server 2>/dev/null || dpkg -l | grep telnetd)

if [ "$ssh_status" = "active" ] && [ -z "$telnet_installed" ]; then
  echo "U-60,양호,원격 접속 시 SSH 프로토콜만 사용되고 Telnet은 비활성화되어 있습니다."
else
  echo "U-60,취약,SSH 외에 Telnet이 설치되어 있거나 활성화되어 있을 수 있습니다."
  echo "SSH 상태: $ssh_status"
  echo "Telnet 설치 여부: $telnet_installed"
fi
```

```
#[U-61] 3.25 ftp 서비스 확인
ftp_process=$(ps -ef | egrep "vsftpd|proftp" | grep -v grep)

if [ -z "$ftp_process" ]; then
  echo "U-61,양호,FTP 서비스가 비활성화되어 있습니다."
else
  echo "U-61,취약,FTP 서비스가 활성화되어 있습니다."
  echo "$ftp_process"
fi
```

```
#[U-62] 3.26 ftp 계정 shell 제한
ftp_shell=$(grep "^ftp:" /etc/passwd | awk -F: '{print $7}')

if [ "$ftp_shell" = "/bin/false" ]; then
  echo "U-62,양호,ftp 계정에 로그인 불가 쉘이 설정되어 있습니다."
else
  echo "U-62,취약,ftp 계정에 로그인 가능한 쉘이 설정되어 있습니다. ($ftp_shell)"
fi
```

```
#[U-63] 3.27 ftpusers 파일 소유자 및 권한 설정
check_file() {
  file_path="$1"

  if [ -f "$file_path" ]; then
    owner=$(stat -c "%U" "$file_path")
    perm=$(stat -c "%a" "$file_path")

    if [ "$owner" = "root" ] && [ "$perm" -le 640 ]; then
      echo "U-63,양호,$file_path의 소유자와 권한이 적절합니다."
    else
      echo "U-63,취약,$file_path의 소유자($owner) 또는 권한($perm)이 부적절합
니다."
    fi
  else
    echo "U-63,양호,$file_path 파일이 존재하지 않습니다."
  fi
```

```
}

check_file "/etc/ftpusers"
check_file "/etc/ftpd/ftpusers"
```

```
#[U-64] 3.28 ftpusers 파일 설정(FTP 서비스 root 계정 접근제한)
FTP_USERS_FILES="/etc/ftpusers /etc/ftpd/ftpusers /etc/vsftp/ftpusers /et
c/vsftp/user_list /etc/vsftpd.ftpusers /etc/vsftpd.user_list"
PROFTPD_CONF="/etc/proftpd.conf"

ROOT_ALLOWED=0
FILE_EXISTED=0

for file in $FTP_USERS_FILES; do
    if [ -f "$file" ]; then
        FILE_EXISTED=1
        if grep -vE '^\s*#' "$file" | grep -qw root; then
            echo "[!] $file 에서 root 계정 허용됨"
            ROOT_ALLOWED=1
        fi
    fi
done

if [ -f "$PROFTPD_CONF" ]; then
    FILE_EXISTED=1
    if grep -i "^RootLogin" "$PROFTPD_CONF" | grep -qw "on"; then
        echo "[!] $PROFTPD_CONF 에서 RootLogin on 설정됨"
        ROOT_ALLOWED=1
    fi
fi

if [ "$FILE_EXISTED" -eq 0 ]; then
    echo "U-64,양호,FTP 관련 설정 파일이 존재하지 않거나, FTP 서비스 미설치"
elif [ "$ROOT_ALLOWED" -eq 0 ]; then
    echo "U-64,양호,FTP 서비스에서 root 계정의 접근이 제한됨"
else
```

```
    echo "U-64,취약,FTP 서비스에서 root 계정의 접근이 허용됨 (접근 제한 필요)"
fi
```

```
#[U-65] 3.29 at 서비스 권한 설정
AT_ALLOW_FILE="/etc/at.allow"
AT_DIR="/var/spool/at"

if [ -f "$AT_ALLOW_FILE" ]; then
    if [ ! -s "$AT_ALLOW_FILE" ]; then
        echo "U-65,취약,$AT_ALLOW_FILE 파일이 비어있습니다. 일반사용자 at 명령어 사용 가능"
    else
        echo "U-65,양호,$AT_ALLOW_FILE 파일이 존재하며 일반사용자 at 명령어 사용이 제한됨"
    fi
else
    echo "U-65,취약,$AT_ALLOW_FILE 파일이 존재하지 않아 일반사용자 at 명령어 사용 가능"
fi

if [ -d "$AT_DIR" ]; then
    PERM=$(stat -c "%a" "$AT_DIR")
    if [ "$PERM" -le 640 ]; then
        echo "U-65,양호,$AT_DIR 디렉터리 권한이 640 이하로 설정됨"
    else
        echo "U-65,취약,$AT_DIR 디렉터리 권한이 640 초과로 설정됨"
    fi
else
    echo "U-65,양호,$AT_DIR 디렉터리가 존재하지 않음"
fi
```

```
#[U-66] 3.30 SNMP 서비스 구동 점검
if ps -ef | grep -v grep | grep -q snmpd; then
  echo "U-66,취약,SNMP 서비스가 구동 중입니다."
else
```

```
  echo "U-66,양호,SNMP 서비스가 구동 중이지 않습니다."
fi
```

```
#[U-67] 3.31 SNMP 서비스 Community String의 복잡성 설정
CONF="/etc/snmp/snmpd.conf"
result=0

if [ ! -f "$CONF" ]; then
  echo "U-67,양호,SNMP 설정 파일이 존재하지 않습니다."
  result=0
else
  COMMUNITY_NAMES=$(grep -i 'community' "$CONF" | awk '{print $2}' |
tr '[:upper:]' '[:lower:]')
fi

  if echo "$COMMUNITY_NAMES" | grep -qwE 'public|private'; then
    echo "U-67,취약,SNMP Community 문자열에 'public' 또는 'private'이 포함되
어 있습니다."
    result=1
  else
    echo "U-67,양호,SNMP Community 문자열이 복잡하게 설정되어 있습니다."
    result=0
  fi
```

```
#[U-68] 3.32 로그온 시 경고 메시지 제공
result=0

check_message_file() {
  file=$1
  desc=$2
  if [ ! -f "$file" ]; then
    echo "U-68,취약,$desc 파일이 존재하지 않습니다."
    result=1
  else
    if [ -s "$file" ]; then
      echo "U-68,양호,$desc 파일에 경고 메시지가 설정되어 있습니다."
    else
```

```
      echo "U-68,취약,$desc 파일에 경고 메시지가 설정되어 있지 않습니다."
      result=1
    fi
  fi
}

check_vsftpd_banner() {
  file="/etc/vsftpd/vsftpd.conf"
  if [ ! -f "$file" ]; then
    echo "U-68,취약,vsftpd 설정 파일이 존재하지 않습니다."
    result=1
  else
    if grep -q "^ftpd_banner=" "$file"; then
      echo "U-68,양호,vsftpd 설정에 FTP 경고 메시지가 설정되어 있습니다."
    else
      echo "U-68,취약,vsftpd 설정에 FTP 경고 메시지가 설정되어 있지 않습니다."
      result=1
    fi
  fi
}

check_sendmail_banner() {
  file="/etc/mail/sendmail.cf"
  if [ ! -f "$file" ]; then
    echo "U-68,취약,sendmail 설정 파일이 존재하지 않습니다."
    result=1
  else
    if grep -q "^O SmtpGreetingMessage=" "$file"; then
      echo "U-68,양호,sendmail 설정에 SMTP 경고 메시지가 설정되어 있습니다."
    else
      echo "U-68,취약,sendmail 설정에 SMTP 경고 메시지가 설정되어 있지 않습니
다."
      result=1
    fi
  fi
}

check_named_conf() {
```

```
  file="/etc/named.conf"
  if [ ! -f "$file" ]; then
    echo "U-68,취약,named 설정 파일이 존재하지 않습니다."
    result=1
  else
    echo "U-68,양호,named 설정 파일이 존재합니다."
  fi
}

check_message_file "/etc/motd" "서버 로그인 메시지 (/etc/motd)"
check_message_file "/etc/issue.net" "Telnet 배너 메시지 (/etc/issue.net)"
check_vsftpd_banner
check_sendmail_banner
check_named_conf
```

```
#[U-69] 3.33 NFS 설정파일 접근권한
FILE="/etc/exports"

if [ ! -f "$FILE" ]; then
  echo "U-69,취약,NFS 설정파일이 존재하지 않습니다."
else
  OWNER=$(ls -ld "$FILE" | awk '{print $3}')
  PERM=$(ls -ld "$FILE" | awk '{print $1}')
  PERM_NUM=$(stat -c "%a" "$FILE")

  if [ "$OWNER" = "root" ] && [ "$PERM_NUM" -le 644 ]; then
    echo "U-69,양호,NFS 설정파일 소유자가 root이며 권한이 644 이하로 설정되어
있습니다."
  else
    echo "U-69,취약,NFS 설정파일 소유자 또는 권한이 적절하지 않습니다."
  fi
fi
```

```
#[U-70] 3.34 expn, vrfy 명령어 제한
FILE="/etc/mail/sendmail.cf"

if [ ! -f "$FILE" ]; then
```

```
  echo "U-70,양호,sendmail 설정 파일이 존재하지 않습니다."
else
  if grep -q -E "^(O )?PrivacyOptions.*noexpn" "$FILE" && grep -q -E "^(O
)?PrivacyOptions.*novrfy" "$FILE"; then
    echo "U-70,양호,sendmail 설정에 noexpn, novrfy 옵션이 설정되어 있습니다."
  else
    echo "U-70,취약,sendmail 설정에 noexpn, novrfy 옵션이 설정되어 있지 않습
니다."
  fi
fi
```

```
#[U-71] 3.35 Apache 웹 서비스 정보 숨김
CONF_FILE="/etc/httpd/conf/httpd.conf"
RESULT=0

if [ ! -f "$CONF_FILE" ]; then
  echo "U-71,취약,Apache 설정 파일이 존재하지 않습니다."
  RESULT=1
fi

if grep -qE "^ServerTokens\s+Prod" "$CONF_FILE"; then
  echo "U-71,양호,ServerTokens가 Prod로 설정되어 있습니다."
else
  echo "U-71,취약,ServerTokens가 Prod로 설정되어 있지 않습니다."
  RESULT=1
fi

if grep -qE "^ServerSignature\s+Off" "$CONF_FILE"; then
  echo "U-71,양호,ServerSignature가 Off로 설정되어 있습니다."
else
  echo "U-71,취약,ServerSignature가 Off로 설정되어 있지 않습니다."
  RESULT=1
fi
```

```
#[U-72] 5.2 정책에 따른 시스템 로깅 설정
CONF_FILE="/etc/syslog.conf"
RESULT=0
```

```
if [ ! -f "$CONF_FILE" ]; then
  echo "U-72,취약,syslog 설정 파일이 존재하지 않습니다."
  RESULT=1
fi

check_pattern() {
  pattern=$1
  description=$2
  if grep -qE "^$pattern" "$CONF_FILE"; then
    echo "U-72,양호,$description 설정이 존재합니다."
  else
    echo "U-72,취약,$description 설정이 없습니다."
    RESULT=1
  fi
}

check_pattern "\*.info;mail.none;authpriv.none;cron.none /var/log/message
s" "기본 메시지 로그"
check_pattern "authpriv.\* /var/log/secure" "인증 관련 로그"
check_pattern "mail.\* /var/log/maillog" "메일 로그"
check_pattern "cron.\* /var/log/cron" "크론 로그"0
check_pattern "\*.alert /dev/console" "alert 로그"
check_pattern "\*.emerg \*" "emerg 로그"
```

```
(echo -e '\xEF\xBB\xBF'; cat result.csv) > result_utf8bom.csv
```