

# IR(Inform Relief) System

-시스템 취약점 점검-

팀	명 :	Checker
지도	교수 :	양환석 교수님
팀	장 :	조서연
팀	원 :	김성한 김예리 김미란 이윤지

2021. 11.

중부대학교 정보보호학과

# 목 차

## 1. 서론

1.1 연구 배경 .....	4
1.2 연구 필요성 .....	4
1.3 연구 목적 및 주제 선정 .....	5

## 2. 관련 연구

2.1 Linux .....	5
2.2 JAVA .....	5
2.3 JSP .....	5
2.4 SERVLET .....	6
2.5 SPRING .....	6
2.6 HTML .....	6
2.7 CSS .....	6
2.8 JAVA SCRIPT .....	6
2.9 JQUERY .....	7
2.10 MYSQL .....	7
2.11 Shell in a box .....	7
3.1 시스템 구성 .....	7
3.2 프로그램 구성 .....	8
3.2.1 웹 동작 .....	8
3.2.2 관리자 .....	10
3.2.3 클라이언트 .....	13

## 4. 결론

4.1 결론 .....	14
--------------	----

4.2 기대 효과 ..... 14

**5. 별첨**

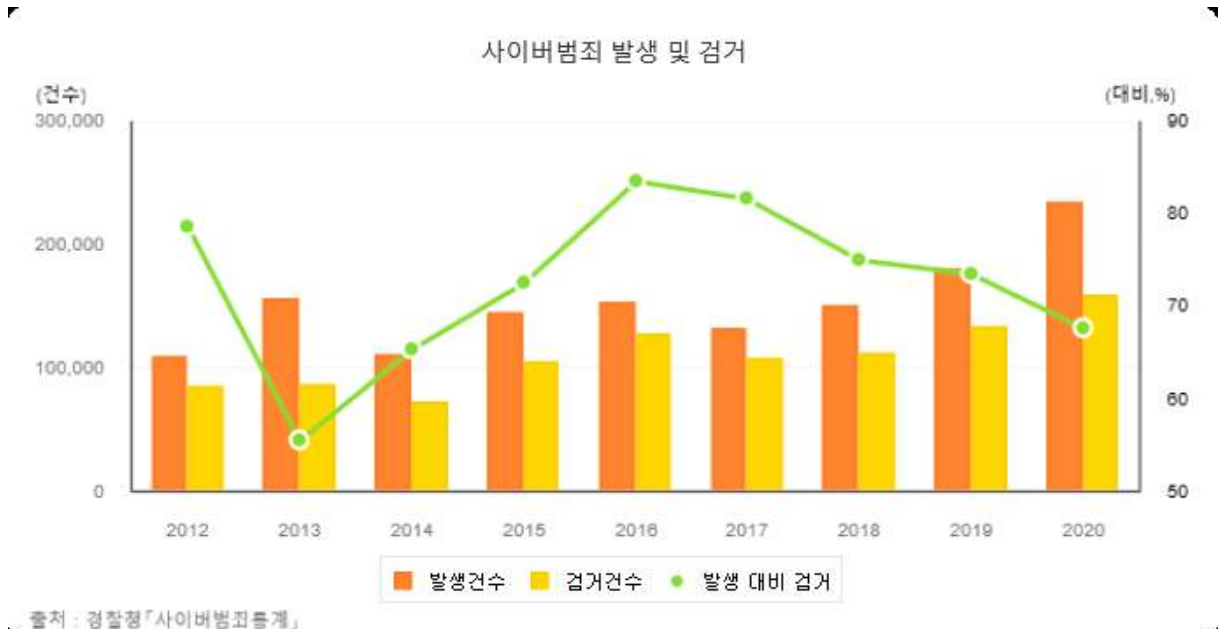
5.1 소스 코드 ..... 14

5.2 발표 자료 ..... 91

# 1. 서론

## 1.1 연구 배경

인터넷 사용인구의 증가로 해킹 및 침해 사고가 빈번히 일어나고 있다. 그림 1은 사이버 범죄 발생 및 검거 그래프로, 그 수가 굉장히 많은 것을 볼 수 있다. 서울시에서 정보 보호의 날을 맞아 발표한 '통계로 본 서울시민의 정보보안 및 인터넷실태'에 따르면 2014년 "정보 보안이 불안하다"는 비율이 65%로 2010년 대비 22.7%가 상승했다고 한다. 또한 2014년에 사이버 범죄 등 "정보보안 관련 피해가 있다"라는 의견은 46.2%로 거의 절반 가까이 차지했고, "피해가 없다"라는 의견은 18.9%로 저조하다. 인터넷 이용자수도 매년 늘어감에 따라 사이버 범죄는 더이상 남의 이야기만은 아니다.



[그림 1] 사이버 범죄 발생 및 검거율

이렇게 매년 시스템 취약점으로 인해 바이러스 및 해킹 사건들이 빈번히 일어나고 있고 이에 따라 시스템을 사용하고 있는 기업들은 경제적 손실을 입고 있으며, 이를 사용하고 있는 소비자들 또한 불안감을 느끼고 있다. 사이버 범죄는 지속적으로 늘어나고 있지만 그에 대응하는 수사전문 인력이 턱없이 부족하고 사이버 범죄의 수준이 점점 높아지면서 대책마련이 시급하다는 목소리가 커지고 있다. 안전한 컴퓨팅 환경을 마련해주는 보안 시스템을 고안하다 보니 보안 취약점 점검 도구 만드는 것이 좋겠다고 생각하여 시스템 취약점 자체를 분석하고 서비스하는 IR시스템을 개발하였다.

## 1.2 연구 필요성

보안설정 사항을 안전하게 유지하는 것은 안전한 시스템을 만드는 최선의 방법인데, 보안 취약점 도구란 바로 이러한 기능을 수행해주는 프로그램을 말한다. 관리자가 일일이 보안 설정을 검사해야 하는 과정을 자동으로 수행함으로써, 서버나 네트워크상의 잠재적 보안 위험요소를 자동으로 진단해주어, 현재의 보안 상태에서 발생 가능한 문제점을 해결할 수 있도록 하는 것이 보안 취약점 점검 기능의 정의이자 목적이다. 시스템 설정 또는

서비스를 통해 공격자에 의해 공격당하지 않도록 보안점검 항목을 점검하여 취약점을 예방할 수 있기때문에 취약점 점검은 꼭 필요하다.

### 1.3 연구 목적 및 주제 선정

2017 '주요정보통신기반시설 기술적 취약점 분석 평가 방법 상세가이드'를 기반으로 웹 페이지에서 SSH접속을 통해 시스템 보안 취약점을 점검하여 발견된 문제들의 상세 원인 분석부터 해결방법까지 보고서를 통해 다운로드 받을 수 있도록 클라이언트 관점에서 간단하게 취약점 점검이 가능하도록 하게 주제를 선정해보았다.

## 2. 관련 연구

### 2.1 LINUX

Linux는 UNIX와 유사하게 설계되었으나 발전을 거듭하며 전화기에서 슈퍼컴퓨터에 이르는 다양한 하드웨어에서 실행되고 있다. 모든 Linux 기반 OS에는 하드웨어 리소스를 관리하는 Linux 커널과 OS의 나머지를 구성하는 일련의 소프트웨어 패키지가 포함되어 있다. OS에는 GNU 툴과 같은 일부 공통 핵심 구성 요소가 포함되어 있다. 이러한 툴을 사용하여 커널에서 제공하는 리소스를 관리하고 추가 소프트웨어를 설치하여 성능 및 보안 환경을 설정할 수 있다. 이러한 모든 툴이 결합되어 기능적인 운영 체제를 구성한다.

### 2.2 JAVA

썬 마이크로시스템즈에서 1995년에 개발한 객체 지향 프로그래밍 언어. 창시자는 제임스 고슬링이다. 2010년에 오라클이 썬 마이크로시스템즈를 인수하면서 Java의 저작권을 소유하였다. 현재는 OpenJDK는 GPL2이나 오라클이 배포하는 Oracle JDK는 상업라이선스로 2019년 1월부터 유료화정책을 강화하고 있다. Java EE는 이클립스 재단의 소유이다. Java 언어는 J2SE 1.4부터는 Java Community Process (JCP)에서 개발을 주도하고 있다. C#과 문법적 성향이 굉장히 비슷하며, 그에 비해 2019년 Q3에서 가장 많이 이용하는 언어로 뽑혔다.

### 2.3 JSP

Java를 이용한 서버 사이드 스크립트 언어. Java Server Pages의 약자이며, 오라클에서 자바상표권 문제로 오픈소스인 jsp는 자카르타 서버페이지로 이름을 바꾸었다. Java의 점유율을 대폭 상승시킨 1등공신이다. 같은 부류에 속하는 것으로 PHP, ASP가 있다. 확장자는 당연히 .jsp를 사용. ASP와 마찬가지로 <% ... %>로 둘러싸인 스크립트 영역이 있으며, 실행시에 javax.servlet.http.HttpServlet 클래스를 상속받은 Java 소스 코드로 변환한 다음 컴파일되어 실행된다. 이 JSP 파일을 Servlet 클래스로 변환하고 실행시켜 주는 역할을 하는 프로그램은 서블릿 컨테이너라고 부른다. 대표적인 것으로 오픈 소스 웹 컨테이너인 톰캣이 있다. 하나의 JSP 페이지가 하나의 Java 클래스이기 때문에 모든 Java 라이브러리를 끌어다 쓸 수 있다.

## 2.4 SERVLET

자바 서블릿(Java Servlet)은 자바를 사용하여 웹페이지를 동적으로 생성하는 서버측 프로그램 혹은 그 사양을 말하며, 흔히 "서블릿"이라 불린다. 자바 서블릿은 웹 서버의 성능을 향상하기 위해 사용되는 자바 클래스의 일종이다. 서블릿은 JSP와 비슷한 점이 있지만, JSP가 HTML 문서 안에 Java 코드를 포함하고 있는 반면, 서블릿은 자바 코드 안에 HTML을 포함하고 있다는 차이점이 있다.

## 2.5 SPRING

스프링 프레임워크(Spring Framework)는 자바 플랫폼을 위한 오픈 소스 애플리케이션 프레임워크로서 간단히 스프링(Spring)이라고도 한다. 동적인 웹 사이트를 개발하기 위한 여러 가지 서비스를 제공하고 있다. 대한민국 공공기관의 웹 서비스 개발 시 사용을 권장하고 있는 전자정부 표준프레임워크의 기반 기술로서 쓰이고 있다.

## 2.6 HTML

하이퍼 텍스트 마크업 언어(Hyper Text Markup Language, HTML, 문화어: 초본문표식달기 언어, 하이퍼본문표식달기언어)는 웹 페이지를 위한 지배적인 마크업 언어다. 또한, HTML은 제목, 단락, 목록 등과 같은 본문을 위한 구조적 의미를 나타내는 것뿐만 아니라 링크, 인용과 그 밖의 항목으로 구조적 문서를 만들 수 있는 방법을 제공한다. 그리고 이미지와 객체를 내장하고 대화형 양식을 생성하는 데 사용될 수 있다. HTML은 웹 페이지 콘텐츠 안의 꺾쇠 괄호에 둘러싸인 "태그"로 되어있는 HTML 요소 형태로 작성한다. HTML은 웹 브라우저와 같은 HTML 처리 장치의 행동에 영향을 주는 자바스크립트와 본문과 그 밖의 항목의 외관과 배치를 정의하는 CSS 같은 스크립트를 포함하거나 불러올 수 있다.

## 2.7 CSS

종속형 시트 또는 캐스케이딩 스타일 시트(Cascading Style Sheets, CSS)는 마크업 언어가 실제 표시되는 방법을 기술하는 스타일 언어(style sheet language)로, HTML과 XHTML에 주로 쓰이며, XML에서도 사용할 수 있다. 기본 파일 명 W3C의 표준이며, 레이아웃과 스타일을 정의할 때의 자유도가 높다. 기본 파일명은 style.css이다. 마크업 언어(ex: HTML)가 웹사이트의 몸체를 담당한다면 CSS는 옷과 액세서리처럼 꾸미는 역할을 담당한다고 할 수 있다. 즉, HTML 구조는 그대로 두고 CSS 파일만 변경해도 전혀 다른 웹사이트처럼 꾸밀 수 있다.

## 2.8 JAVASCRIPT

모질라 재단의 프로토타입기반의 프로그래밍 언어로, 스크립트 언어에 해당된다. 특수한 목적이 아닌 이상 모든 웹 브라우저에 인터프리터가 내장되어 있다. 오늘날 HTML, CSS와 함께 웹을 구성하는 요소 중 하나다. HTML이 웹 페이지의 기본 구조를 담당하고, CSS가 디자인을 담당한다면 JavaScript는 클라이언트 단에서 웹 페이지가 동작하는 것을 담당한다.<sup>[1]</sup> 웹 페이지를 자동차에 비유하자면, HTML은 자동차의 뼈대, CSS는 자동차의 외관, JavaScript는 자동차의 동력이라고 볼 수 있다. 또한 Node.js와 같은 런타임 환경과 같이 서버 프로그래밍에도 사용되고 있다.

## 2.9 JQUERY

jQuery(제이쿼리)는 HTML의 클라이언트 사이드 조작을 단순화 하도록 설계된 크로스 플랫폼의 자바스크립트 라이브러리다. 존 레식이 2006년 뉴욕 시 바캠프(Barcamp NYC)에서 공식적으로 소개하였다. jQuery는 MIT 라이선스를 가진 자유 오픈 소프트웨어이다. jQuery의 문법은 코드 보기, 문서 객체 모델 찾기, 애니메이션 만들기, 이벤트 제어, Ajax 개발을 쉽게 할 수 있도록 디자인되었다. 또한, jQuery는 개발자가 플러그인을 개발할 수 있는 기능을 제공한다.

## 2.10 MYSQL

MYSQL(마이에스큐엘)은 세계에서 가장 많이 쓰이는 오픈 소스의 관계형 데이터베이스 관리 시스템(RDBMS)이다. 다중 스레드, 다중 사용자 형식의 구조질이어 형식의 데이터베이스 관리 시스템으로서 오라클이 관리 및 지원하고 있으며, Qt처럼 이중 라이선스가 적용된다. 하나의 옵션은 GPL이며, GPL 이외의 라이선스로 적용시키려는 경우 전통적인 지적재산권 라이선스의 적용을 받는다.

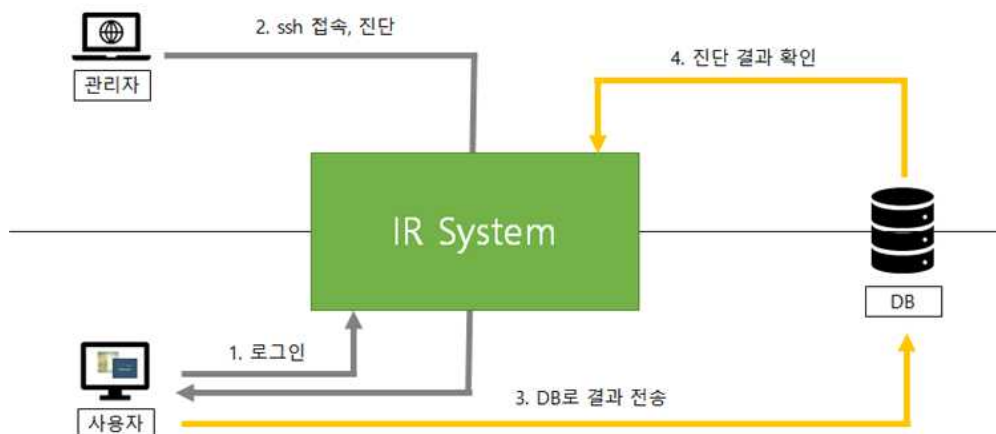
## 2.11 Shell in a box

Shell in a box는 Markus Gutschke가 만든 웹 기반 터미널 에뮬레이터다. 지정된 포트에서 웹 기반 SSH 클라이언트로 실행되고 웹 터미널 에뮬레이터가 필요없이 AJAX/JavaScript 및 CSS 지원 브라우저를 사용하여 원격으로 Linux 서버 SSH 셸에 액세스하고 제어하도록 프롬프트하는 내장 웹 서버가 존재한다. 웹 기반 SSH는 방화벽으로 보호되고 HTTP(s) 트래픽만 통과할 수 있는 경우에 매우 유용하다.

## 3. 본론

### 3.1 시스템 구성

리눅스 운영체제 기반으로 만들었다.

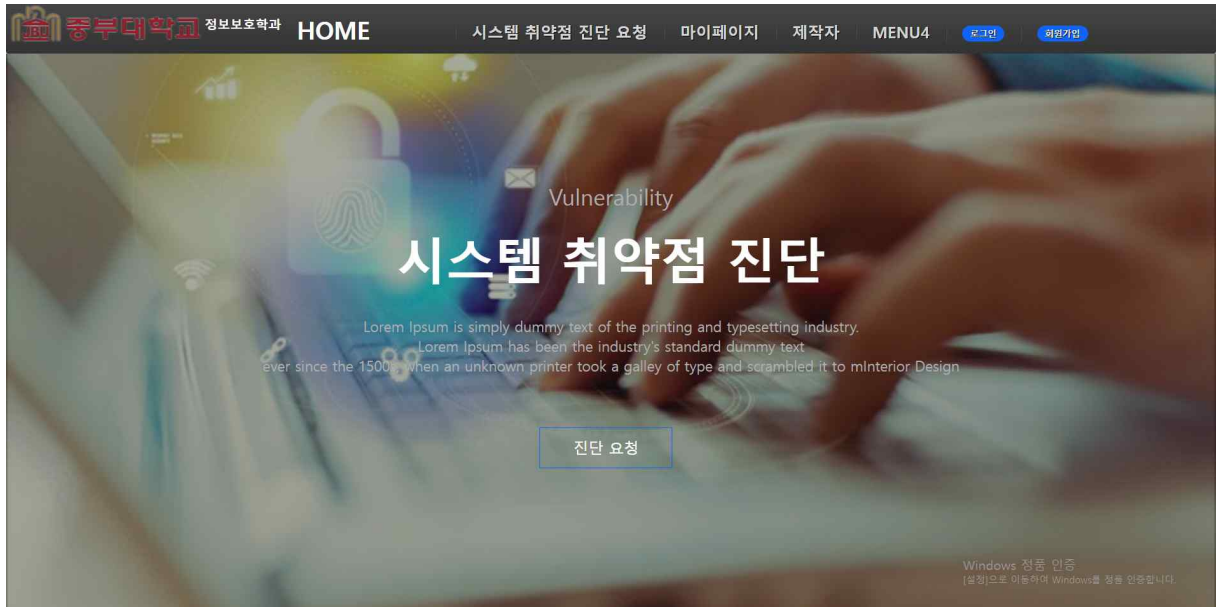


[그림 2] 프로그램 제작 구성도

## 3.2 프로그램 구성

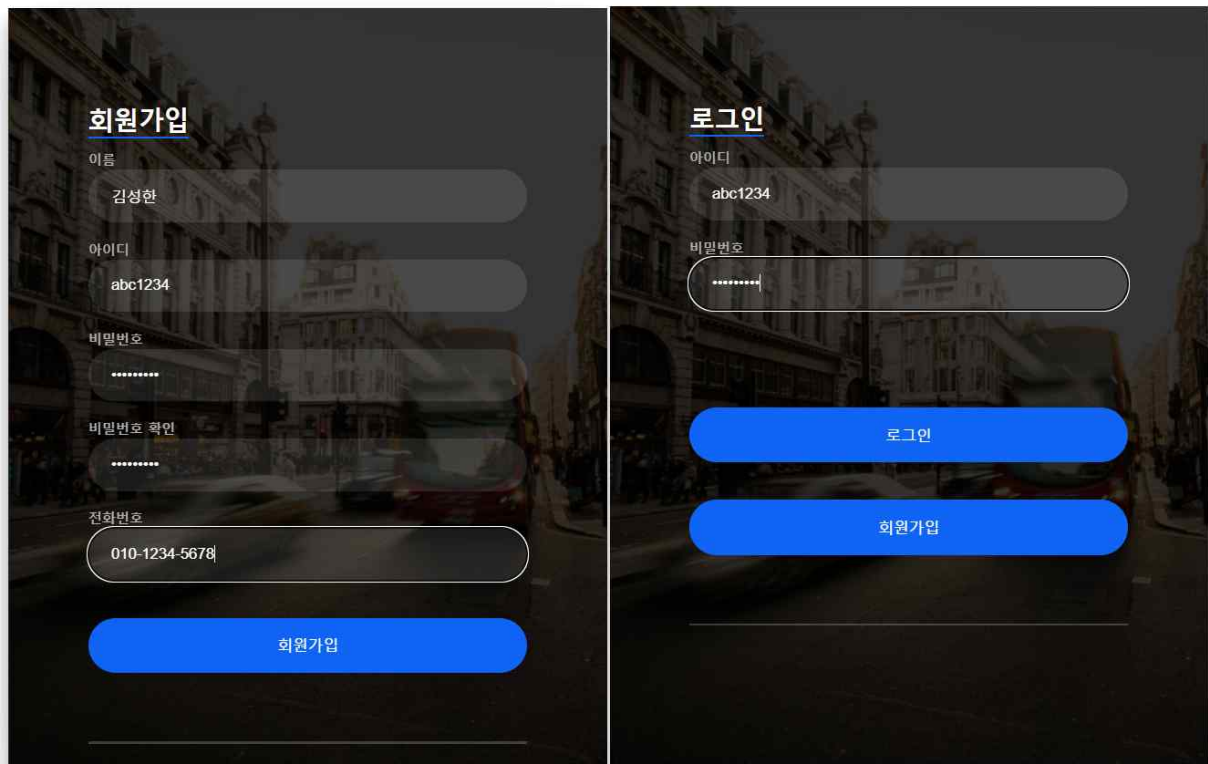
### 3.2.1 웹

취약점 진단을 받고 싶은 클라이언트는 먼저 웹페이지에 접속한다.



[그림 3] 메인 페이지

취약점 점검을 위해 회원가입을 한 후, 로그인을 한다.

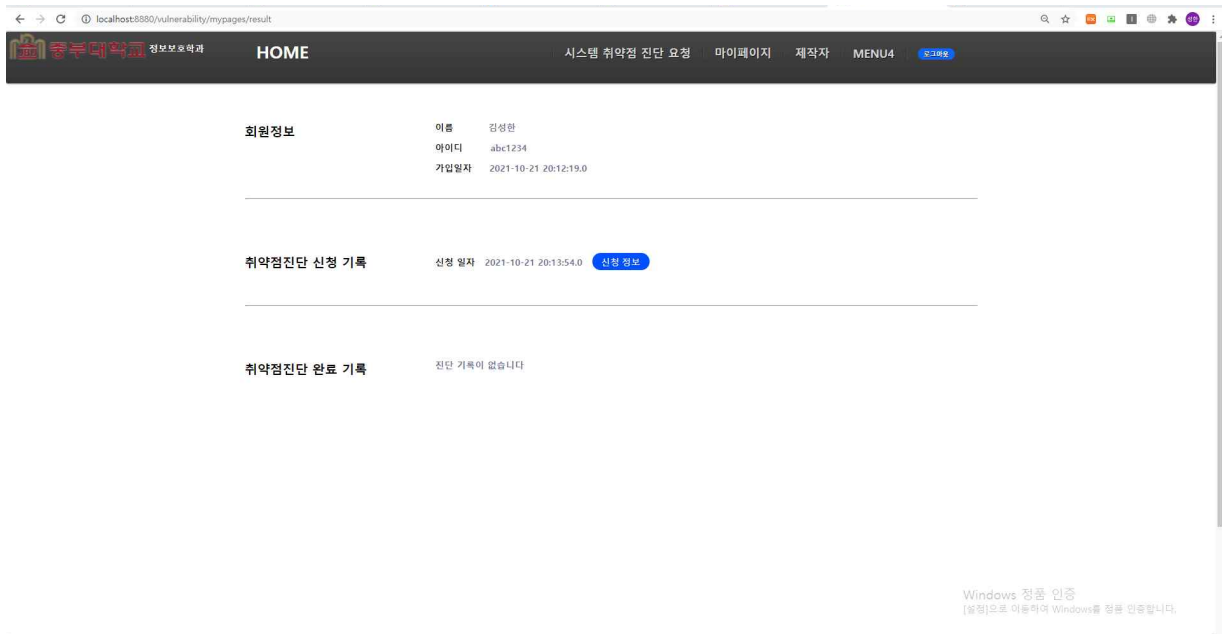


[그림 4] 회원가입 및 로그인



취약점 진단요청 페이지로 넘어간다.

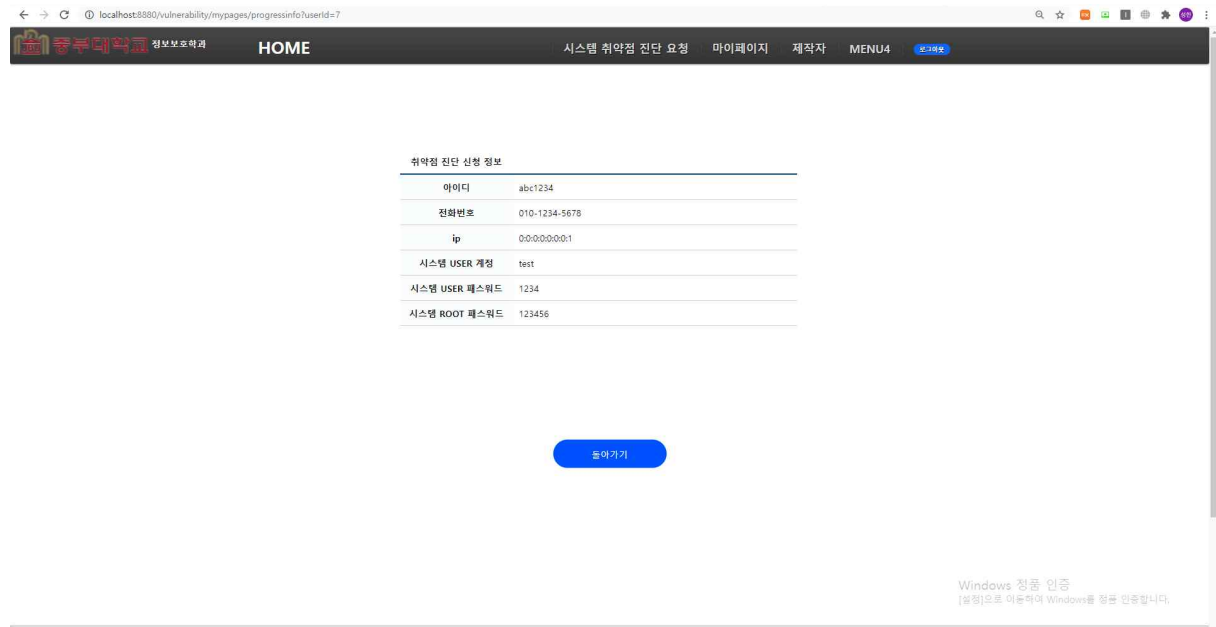
[그림 5] 진단요청 페이지



[그림 6] 마이페이지

관리자가 ssh로 원격접속을 하기위해서는 진단요청 페이지에서 유저계정 이름과 시스템 비밀번호 그리고 시스템의 root 비밀번호가 필요하기때문에 그림 5와 같이 필요한 정보

들을 진단요청 페이지에서 입력받는다.



[그림 7] 마이페이지 - 신청기록 정보

마이페이지에서 본인이 신청한 정보를 확인 할 수 있다.

### 3.2.2 관리자



[그림 8] 관리자 페이지

## 취약점 진단 신청 정보

아이디	abc1234
전화번호	010-1234-5678
ip	0.0.0.0:0:1
시스템 USER 계정	test
시스템 USER 패스워드	1234
시스템 ROOT 패스워드	123456

진단하기

[그림 9] 관리자 페이지 - 진단하기

```

root@centos7:/opt/IR - Shell In A Box - Chrome
주의 요함 | https://172.20.113.50:4200
[root@centos7 opt]# pwd
/opt
[root@centos7 opt]# ls
IR.tar rh
[root@centos7 opt]# tar -xvf IR.tar
IR/
IR/result.sh
IR/account.sh
IR/file.sh
IR/service.sh
IR/all.sh
IR/final_DB.sh
IR/function.sh
[root@centos7 opt]# ls
IR IR.tar rh
[root@centos7 opt]# cd IR
[root@centos7 IR]# ls
account.sh file.sh function.sh service.sh
all.sh final_DB.sh result.sh
[root@centos7 IR]#

```

[그림 10] shell in a box 원격 접속

그림 9에서 진단하기 버튼을 누르면 그림 10에서와 같이 shell in a box를 이용하여 원격으로 웹을 통해 점검대상의 리눅스에 접속한다.

```

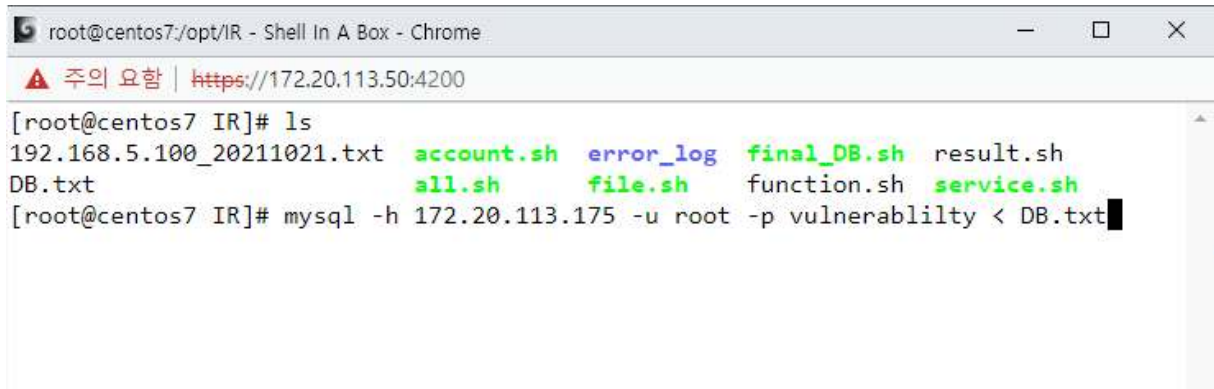
root@centos7:/opt/IR - Shell In A Box - Chrome
주의 요함 | https://172.20.113.50:4200
[root@centos7 IR]# ./all.sh
취약점 점검 시작
[##]10%

root@centos7:/opt/IR - Shell In A Box - Chrome
주의 요함 | https://172.20.113.50:4200
[root@centos7 IR]# ./all.sh
취약점 점검 시작
[#####]100%
취약점 점검 완료!
[root@centos7 IR]# ls
192.168.5.100_20211021.txt  error_log  function.sh
account.sh                file.sh    result.sh
all.sh                    final_DB.sh service.sh
[root@centos7 IR]# ./final_DB.sh
FILE NAME: 192.168.5.100_20211021.txt
USER ID: 2
complete!
[root@centos7 IR]# ls
192.168.5.100_20211021.txt  account.sh  error_log  final_DB.sh  result.sh
DB.txt                    all.sh     file.sh    function.sh  service.sh
[root@centos7 IR]#

```

[그림 11,12] 취약점 점검 시작 및 점검완료 후 DB가공

all.sh라는 스크립트를 실행하여 모든항목을 점검하면 사용자 아이피 +점검한 날짜로된 결과파일이 생성된다. 후에 final\_DB를 돌려 DB.txt 파일을 생성한다.



[그림 13] DB에 밀어 넣기

진단결과를 표의 형식으로 가공하기 위해 DB.txt파일을 DB에 밀어 넣어준다.

### 3.2.3 클라이언트



[그림 14] 점검결과 확인

진단 결과를 상세결과표 형식으로 확인 할 수 있다.

## 4. 결론

### 4.1 결론

기존의 취약점 진단시스템들과 달리 진단결과를 만족, 부분만족, 불만족 이렇게 3가지로 구분 지었다. 어떤 항목안에 꼭 하나의 점검이 아닌 여러 점검사항들이 존재할 경우를 대비한 것이다. 여러 점검 사항 중 모든 사항이 양호할 경우 만족, 어떤 사항은 양호하지만 어떤 사항은 취약할 경우 부분만족 그리고 모든 사항이 취약하면 불만족으로서 진단 결과가 나오도록 했다. 또한 각 항목별로 단순히 취약한지 양호한지 점검하는 것이 아닌 취약한 부분이 있어서 부분만족 혹은 불만족이 뜨면 어떤 부분이 문제였는지 조치사항까지 나오도록 하였다.

### 4.2 기대효과

IR시스템은 보안상의 위협을 급감시킬 수 있고, 관리자가 효율적으로 시스템을 안전한 상태로 유지 시킬 수 있다. 이외에 IR시스템이 갖는 부가적인 그래픽 인터페이스를 통한 편리한 사용성과 사용자가 보기 편하도록 만들어진 진단결과 등을 들 수 있다. 이 IR시스템을 도입하면 주요기반시설, 금융기반 시설 등 취약점 점검과 관련된 법적 보안 규제 준수를 만족하게 되고, 주기적인 취약점 점검을 통한 보안사고에 대한 사전예방을 강화할 수 있다. 또한 시스템 관리자의 업무효율 증대와 보안의식 그리고 보안수준의 향상을 도모 할 수 있다.

## 5. 별첨

### 5.1 소스코드

#### function.sh

```
#!/bin/bash
. result.sh
MAIN_PATH=`dirname $0`
mkdir ${MAIN_PATH}/error_log > /dev/null 2>&1
ERROR_PATH="${MAIN_PATH}/error_log"
U-01()
{echo ""
echo "[분류]"
echo "계정관리"
echo ""
echo "[항목코드]"
echo "U-01"
echo ""
echo "[중요도]"
```

```

echo "상"
echo ""
echo "[점검항목]"
echo "root 계정 원격 접속 제한"
echo ""
echo "[점검현황]"
####
#[telnet 서비스 사용시]
telnet=`grep pts /etc/securetty 2>${ERROR_PATH}/error_01`
if [ $? -eq 0 ];then
    echo ${telnet} | grep '^#'
    if [ $? -eq 0 ];then
        echo "1 /etc/securetty 파일 내 *pts/x 관련 설정이 주석 처리 되어있음" >>
result.txt
    else
        echo "0 /etc/securetty 파일 내 *pts/x 관련 설정이 존재함" >> result.txt
        echo "PAM 모듈 설정과 관계 없이 root 계정 접속을 허용하므로 반드시
securetty 파일에서 pts/x 관련 설정 제거 필요" >> action.txt
    fi
else
    echo "1 /etc/securetty 파일 내 *pts/x 관련 설정이 존재하지 않음" >> result.txt
fi
#[SSH 서비스 사용시]
ssh=`sed -n -e "/Authentication:./,/PermitRootLogin/p" /etc/ssh/sshd_config
2>${ERROR_PATH}/error_01`
option=`echo ${ssh}#*PermitRootLogin`
#옵션 뒤의 문자열 추출
comment=`echo "${ssh}" | grep PermitRootLogin | grep "^#" `
if [ $? -eq 0 ];then
    echo "0 PermitRootLogin에 주석 설정이 되어 있음" >> result.txt
    echo "PermiRootLogin 설정에서 주석을 제거한 후 No로 설정" >> action.txt
elif [[ ${option} == "yes" || ${option} == "Yes" ]];then
    echo "0 PermitRootLogin설정이 되어 있음" >> result.txt
    echo "PermiRootLogin설정을 No로 변경" >> action.txt
elif [[ ${option} == "no" || ${option} == "No" ]];then
    echo "1 PermitRootLogin설정이 No로 설정되어 있음" >> result.txt
else
    error=`cat ${ERROR_PATH}/error_01`
    echo "2 에러 메시지: "${error}" >>result.txt
fi
detail
###

```

```

echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-02()
{echo ""
echo "[분류]"
echo "계정관리"
echo ""
echo "[항목코드]"
echo "U-02"
echo ""
echo "[점검항목]"
echo "패스워드 복잡성 설정"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
####
pwd="/etc/security/pwquality.conf"
echo -e "ucredit:-1Wndcredit:-1Wnocredit:-1Wnminlen:8Wndifok:N" >>
U-02_comparison_list.txt
for file in `cat U-02_comparison_list.txt`
do
    com_optname=`echo ${file} | cut -f 1 -d ':'`
    com_optvalue=`echo ${file} | cut -f 2 -d ':'`
    optname=`grep ${com_optname} $pwd 2>${ERROR_PATH}/error_02 `
    optvalue=`echo ${optname#*=} 2>${ERROR_PATH}/error_02`
    cat $pwd | grep '^#' | grep ${com_optname}>> /dev/null
    if [ $? -eq 0 ];then
        echo "0 ${com_optname} 설정에 주석처리 되어 있음">>result.txt
        echo "${com_optname} 값이 ${com_optvalue}(이)가 되게 설정해야함"
>>action.txt
        elif [ ${optvalue} != ${com_optvalue} ];then
            echo "0 ${com_optname} 설정이 요구사항과 일치하지 않음">>result.txt
            echo "${com_optname} 값이 ${com_optvalue}(이)가 되게 설정해야함"
>>action.txt
        elif [ ${optvalue} == ${com_optvalue} ];then
            echo "1 ${com_optname} 설정이 요구사항과 일치함">>result.txt

```



```

else
    error=`cat $ERROR_PATH/error_02`
    echo "2 에러 메시지: "${error}" " >> result.txt
fi
done
rm -rf U-02_comparison_list.txt
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action >> act.txt
sort -u act.txt
rm -rf act.txt > /dev/null 2>&1
}U-03()
{echo ""
echo "[분류]"
echo "계정관리"
echo ""
echo "[항목코드]"
echo "U-03"
echo ""
echo "[점검항목]"
echo "계정 잠금 임계값 설정"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
####
pam=`cat /etc/pam.d/system-auth 2>$ERROR_PATH/error_03`
pam_module=pam_faillock.so
#pam_module=pam_tally.so
#pam_module=pam_tally2.so
#위에 설정 값이 버전에 따라 달라지기 때문에 cat /etc/*release* 명령어로 버전 확인 후 진행하기
#if 7버전이면 faillock.so 모듈을 이용할 수 있도록
#centos 7 버전에서는 faillock.so와 pam_tally2.so는 정상 작동함
#경로 : /lib64/security/pam_faillock.so, /lib64/security/pam_tally2.so
echo ${pam} |grep '^auth' | grep $pam_module

```

```

if [ $? -ne 0 ];then
    echo "0 계정 임계값이 설정되어 있지 않음" >> result.txt
    echo "아래와 같은 내용으로 변경" >> action.txt
    echo "auth required /lib64/security/pam_faillock.so deny=5 unlock_time=120
no_magic_root" >> action.txt
    echo "account required /lib/security/pam_tally.so no_magic_root reset" >> action.txt
else
    echo "1 계정 임계값이 설정되어 있음" >> result.txt
fi
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-04()
{echo ""
echo "[분류]"
echo "계정관리"
echo ""
echo "[항목코드]"
echo "U-04"
echo ""
echo "[점검항목]"
echo "패스워드 파일 보호"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
####
file="/etc/shadow"
if [ -f $file ];then
    echo "1 /etc/shadow 파일이 존재함" >> result.txt
else
    echo "0 /etc/shadow 파일이 존재하지 않음" >> result.txt
    echo "패스워드 암호화 저장 및 관리 설정 적용 필요" >> action.txt
fi
# /etc/passwd 파일 내 두번째 필드가 x로 표시되는지 확인
passwd_file=`cat /etc/passwd | grep '^root' $1 | awk -F: '{print $2}`

```

```

if [ $? -eq 0 ];then
    if [ ${passwd_file} == "x" ];then
        echo "1 /etc/passwd 파일 내 두번째 필드가 x임" >> result.txt
    else
        echo "0 /etc/passwd 파일 내 두번째 필드가 x가 아님" >> result.txt
        echo "/etc/shadow 파일 내 비밀번호가 암호화된 형태로 저장되어 있는지
확인 필요" >> action.txt
    fi
fi
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-05()
{echo ""
echo "[분류]"
echo "파일 및 디렉터리 관리"
echo ""
echo "[항목코드]"
echo "U-05"
echo ""
echo "[점검항목]"
echo "root 홈, 패스 디렉터리 권한 및 패스 설정"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
echo $PATH > /dev/null 2>&1
if [ $? = 0 ]; then
    TMP=1
fi
#기능 구현
A_code=`echo $PATH`
if [ $TMP == 1 ]; then
    if [[ $A_code == *.*.* || $A_code == *:*.*.* ]]; then
        echo 0 PATH 환경변수에 마침표가 맨 앞이나 중간에 포함되어 있음 >>
result.txt

```

```

        echo root 계정의 환경변수 설정파일에서 PATH 환경변수에 포함되어 있는
현재 디렉터리를 나타내는 마침표를 PATH 환
경변수의 마지막으로 이동 >> action.txt
    else
        echo 1 PATH 환경변수에 마침표가 맨 앞이나 중간에 포함되지 않음 >>
result.txt
    fi
elif [ $TMP == 0 ]; then
    echo 2 환경변수가 설정되어 있지 않음 >> result.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-06()
{echo ""
echo "[분류]"
echo "파일 및 디렉터리 관리"
echo ""
echo "[항목코드]"
echo "U-06"
echo ""
echo "[점검항목]"
echo "파일 및 디렉터리 소유자 설정"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
####
find / -nouser -print > /dev/null 2>&1
touch result.txt
if [ $? -eq 1 ];then
    echo "0 소유자가 nouser인 파일이나 디렉터리가 존재함" >> reuslt.txt
    echo "소유자가 존재하지 않는 파일 및 디렉터리 삭제 또는, 소유자 변경" >>
action.txt
else
    echo "1 소유자가 nouser인 파일이나 디렉터리가 존재하지 않음" >> result.txt
fi

```

```

find /-nogroup -print > /dev/null 2>&1
if [ $? -eq 0 ];then
    echo "0 소유자가 nogroup인 파일이나 디렉터리가 존재함" >> result.txt
    echo "소유자가 존재하지 않는 파일 및 디렉터리 삭제 또는, 소유자 변경" >>
action.txt
fi
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-07()
{echo ""
echo "[분류]"
echo "파일 및 디렉터리 관리"
echo ""
echo "[항목코드]"
echo "U-07"
echo ""
echo "[점검항목]"
echo "/etc/passwd 파일 소유자 및 권한 설정"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
####
touch result.txt
chpwd=`ls -l /etc/passwd | awk '{print $3}`
if [ ${chpwd}='root' ];then
    echo "1 /etc/passwd의 소유자가 root임" >> result.txt
else
    echo "0 /etc/passwd의 소유자가 root가 아님" >> result.txt
    echo "/etc/passwd의 소유자를 root로 변경" >> action.txt
fi
pam=`ls -l /etc/passwd | awk -F . '{ print $1 }'`
pam1=`echo ${pam} | cut -c2-4 `
pam2=`echo ${pam} | cut -c5-7 `
pam3=`echo ${pam} | cut -c8-11`

```

```

echo "현재 /etc/passwd 의 권한 : ${pam}"
if [ ${pam1} == "rwx" ];then
    echo -e "0 644이하로 설정되어 있지 않음" >> result.txt
    echo "644이하로 설정해야 함" >> action.txt
elif [ ${pam1} == "rw-" ];then
    if [ ${pam2} == "rwx" ] || [ ${pam2} == "rw-" ] || [ ${pam2} == "r-x" ];then
        echo "0 644이하로 설정되어 있지 않음" >> result.txt
    elif [ ${pam3} == "rwx" ] || [ ${pam3} == "rw-" ] || [ ${pam3} == "r-x" ];then
        echo "0 644이하로 설정되어 있지 않음" >> result.txt
    else
        echo "1 644이하로 설정 되어 있음" >> result.txt
    fi
else
    echo "1 644이하로 설정 되어 있음" >> result.txt
fi
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-08()
{echo ""
echo "[분류]"
echo "파일 및 디렉터리 관리"
echo ""
echo "[항목코드]"
echo "U-08"
echo ""
echo "[점검항목]"
echo "/etc/shadow 파일 소유자 및 권한 설정"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
####
TMP=`ls -l /etc/shadow | awk '{print $3}'`
if [ ${TMP}=='root' ];then
echo "1 /etc/shadow의 소유자는 root임 " >> result.txt

```

```

else
echo "0 /etc/shadow의 소유자는 root가 아님" >> result.txt
echo "/etc/shadow의 소유자 root로변경" >> action.txt
fi
TMP2=`ls -l /etc/shadow | awk '{print$1}`
if [ ${TMP2}=='-r-----' ];then
echo "1 /etc/shadow의 권한이 400으로 설정 되어있음" >> result.txt
else
echo "0 /etc/shadow의 권한이 400으로 설정 되어있지 않음" >> result.txt
echo "/etc/shadow의 권한을 400으로 변경" >> action.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-09()
{echo ""
echo "[분류]"
echo "파일 및 디렉터리 관리"
echo ""
echo "[항목코드]"
echo "U-09"
echo ""
echo "[점검항목]"
echo "/etc/hosts 파일 소유자 및 권한 설정"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
####
TMP=`ls -l /etc/hosts | awk '{print $3}`
if [ ${TMP}=='root' ];then
echo "1 /etc/hosts의 소유자는 root임 " >> result.txt
else
echo "0 /etc/hosts의 소유자는 root가 아님" >> result.txt
echo "/etc/hosts의 소유자 root로변경" >> action.txt
fi
pam=`ls -l /etc/hosts | awk -F . '{print $1}`

```

```

pam1=`echo ${pam} | cut -c2-4 `
pam2=`echo ${pam} | cut -c5-7 `
pam3=`echo ${pam} | cut -c8-11 `
echo "현재 /etc/hosts 파일 권한 : ${pam}"
if [ ${pam1} == 'rwx' ];then
    echo "0 /etc/hosts 권한이 600이상임" >> result.txt
    echo "/etc/hohsts 권한이 600이하로 설정되어야 함" >> action.txt
elif [ ${pam2} == "---" ] && [ ${pam3} == "---" ];then
    echo "1 /etc/hosts 권한이 600이하임" >> result.txt
else
    echo "0 /etc/hosts 권한이 600이상임" >> result.txt
    echo "/etc/hohsts 권한이 600이하로 설정되어야 함" >> action.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-10()
{echo ""
echo "[분류]"
echo "파일 및 디렉터리 관리"
echo ""
echo "[항목코드]"
echo "U-10"
echo ""
echo "[점검항목]"
echo "/etc/xinetd.conf 파일 소유자 및 권한 설정"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
###취약점 내용 처리
ls -l /etc/xinetd.conf 2>${ERROR_PATH}/error_10
if [ $? -eq 0 ];then
    chinetd=`ls -l /etc/xinetd.conf | awk '{print $3}'`
    if [ ${chinetd}='root' ];then
        echo "1 /etc/xinetd.conf의 소유자가 root임" >> result.txt
    else

```



```

        echo "0 /etc/xinetd.conf의 소유자가 root가 아님" >> result.txt
        echo "/etc/xinetd.conf의 소유자를 root로 변경" >> action.txt
    fi
else
    error=`cat $ERROR_PATH/error_10`
    echo "2 에러 메시지: ${error}" >> result.txt
fi
inetd=`find /etc/xinetd.conf -perm 600 2>$ERROR_PATH/error_10`
if [ $? -eq 0 ];then
    if [ "${inetd}" == "/etc/xinetd.conf" ];then
        echo "1 /etc/xinetd.conf의 권한이 600임" >> result.txt
    else
        echo "0 /etc/xinetd.conf의 권한이 600이 아님" >> result.txt
        echo "/etc/xinetd.conf의 권한을 600으로 설정" >> action.txt
    fi
else
    error=`cat $ERROR_PATH/error_10`
    echo "2 에러 메시지: ${error}" >> result.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-11()
{echo ""
echo "[분류]"
echo "파일 및 디렉터리 관리"
echo ""
echo "[항목코드]"
echo "U-11"
echo ""
echo "[점검항목]"
echo "/etc/syslog.conf 파일 소유자 및 권한 설정"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
#####

```

```

TMP=`ls -l /etc/rsyslog.conf | awk '{print $3}`
if [ ${TMP}=='root' ];then
echo "1 /etc/rsyslog.conf의 소유자는 root임 " >> result.txt
else
echo "0 /etc/rsyslog.conf의 소유자는 root가 아님" >> result.txt
echo "/etc/rsyslog.conf의 소유자 root로변경" >> action.txt
fi
pam=`ls -l /etc/rsyslog.conf | awk -F . '{ print $1 }'`
pam1=`echo ${pam} | cut -c2-4 `
pam2=`echo ${pam} | cut -c5-7 `
pam3=`echo ${pam} | cut -c8-11`
echo "현재 /etc/rsyslog.conf 의 권한 : ${pam}"
if [ ${pam1} == "rwx" ];then
    echo -e "0 644이하로 설정되어 있지 않음" >> result.txt
    echo "644이하로 설정해야 함" >> action.txt
elif [ ${pam1} == "rw-" ];then
    if [ ${pam2} == "rwx" ] || [ ${pam2} == "rw-" ] || [ ${pam2} == "r-x" ];then
        echo "0 644이하로 설정되어 있지 않음" >> result.txt
    elif [ ${pam3} == "rwx" ] || [ ${pam3} == "rw-" ] || [ ${pam3} == "r-x" ];then
        echo "0 644이하로 설정되어 있지 않음" >> result.txt
    else
        echo "1 644이하로 설정 되어 있음" >> result.txt
    fi
else
    echo "1 644이하로 설정 되어 있음" >> result.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-12()
{echo ""
echo "[분류]"
echo "파일 및 디렉터리 관리"
echo ""
echo "[항목코드]"
echo "U-12"
echo ""
echo "[점검항목]"

```

```

echo "/etc/services 파일 소유자 및 권한 설정"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
chk_services=`ls -l /etc/services 2>$ERROR_PATH/error_10 | awk '{print $1}'`
if [ $? -eq 0 ];then
    echo "/etc/services의 현재 권한 : ${chk_services}"
    chservices=`ls -l /etc/services | awk '{print $3}'`
    if [ ${chservices}=='root' ];then
        echo "1 /etc/services의 소유자가 root임" >> result.txt
    else
        echo "0 /etc/services의 소유자가 root가 아님" >> result.txt
        echo "/etc/services의 소유자를 root로 변경" >> action.txt
    fi
else
    error=`cat $ERROR_PATH/error_10`
    echo "2 에러 메시지: ${error}" >> result.txt
fi
find /etc/services -perm 600 2>$ERROR_PATH/error_10 > 12.txt
if [ $? -eq 0 ];then
    if [ -s 12.txt ];then
        echo "1 /etc/services의 권한이 600임" >> result.txt
    else
        echo "0 /etc/services의 권한이 600이 아님" >> result.txt
        echo "/etc/services의 권한을 600으로 설정" >> action.txt
    fi
else
    error=`cat $ERROR_PATH/error_10`
    echo "2 에러 메시지: ${error}" >> result.txt
fi
rm -rf 12.txt
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-13()
{echo ""

```

```

echo "[분류]"
echo "파일 및 디렉터리 관리"
echo ""
echo "[항목코드]"
echo "U-13"
echo ""
echo "[점검항목]"
echo "SetUID, SetGID, Stickey Bit 설정파일 검사"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
SF="SetUID.txt"
SG="SetGID.txt"
SB="Stickey_Bit.txt"
find / -user root -perm -4000 > $SF > /dev/null 2>&1
find / -user root -perm -2000 > $SG > /dev/null 2>&1
find / -user root -perm -1000 > $SB > /dev/null 2>&1
echo "2 의심스러운 파일 혹은 특이한 파일 확인 후 수동 제거" >> result.txt
echo "불필요한 SUID, SGID 파일 제거" >> action.txt
rm -rf SetUID.txt
rm -rf SetGID.txt
rm -rf Stickey_Bit.txt
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-14()
{echo ""
echo "[분류]"
echo ""파일 및 디렉터리 관리
echo ""
echo "[항목코드]"
echo "U-14"
echo ""
echo "[점검항목]"
echo "사용자, 시스템 시작파일 및 환경파일 소유자 및 권한 설정 "
echo ""

```

```

echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
echo "서버 환경마다 파일들이 다르기 때문에 이를 스크립트로 체크할 경우 오탐 발생이 높음" >> action.txt
echo "관리자의 수동 체크 후, 해당 파일이 root와 소유자만이 w 권한이 있도록 설정" >> action.txt
echo "해당 파일이 root와 소유자 이외에도 w 권한이 있다면 취약하다고 판단할 수 있음" >> action.txt
echo "2 권한 수동 확인 필요" >> result.txt
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-15()
{echo ""
echo "[분류]"
echo "파일 및 디렉터리 관리"
echo ""
echo "[항목코드]"
echo "U-15"
echo ""
echo "[점검항목]"
echo "world writable 파일 점검"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
find / -type f -perm -2 -exec ls -l {} \; > ww.txt /dev/null 2>&1
if [ -e ww.txt ];then
    echo "2 서버 환경마다 다르기 때문에 수동적인 체크가 필요함" >> result.txt
    echo "기본적으로 생성되는 world writable 파일 간의 비교요망" >> action.txt
else
    echo "1 world writable 파일이 존재하지 않음" >> result.txt
fi
rm -rf ww.txt >> /dev/null 2>&1
detail

```

```

#####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-16()
{echo ""
echo "[분류]"
echo "파일 및 디렉터리 관리"
echo ""
echo "[항목코드]"
echo "U-16"
echo ""
echo "[점검항목]"
echo "/dev에 존재하지 않는 device 파일 점검"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
#####
find /dev -type f -exec ls -l {} \; > dev.txt 2>&1
if [ -e dev.txt ];then
echo "2 파일이 존재함" >> result.txt
echo "/dev 파일을 확인요망(수동점검 필요)" >> action.txt
else
echo "1 파일이 존재하지 않음" >> result.txt
fi
rm -rf dev.txt >> /dev/null 2>&1
detail
#####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-17()
{echo ""
echo "[분류]"

```

```

echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-17"
echo ""
echo "[점검항목]"
echo "$HOME/.rhosts, hosts.equiv 사용 금지"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
if [ -f `ls -l $HOME/.rhosts 2>/dev/null` ];then
    if [ -f `ls -l hosts.equiv 2>/dev/null` ];then
        echo "1 해당 서비스가 비활성화 되어있음" >> result.txt
    else
        echo "0 해당 서비스가 활성화 되어있음" >> result.txt
        echo "0 해당 서비스가 활성화 되어있음" >> result.txt
        echo "두 파일의 소유자를 root로 변경 후, 권한 600 이하로 설정" >>
action.txt
        echo "두 파일에서 "+"를 제거하고 반드시 필요한 호스트 및 계정만 등록"
>> action
        fi
    else
        echo "0 해당 서비스가 활성화 되어있음" >> result.txt
        echo "두 파일의 소유자를 root로 변경 후, 권한 600 이하로 설정" >> action.txt
        echo "두 파일에서 "+"를 제거하고 반드시 필요한 호스트 및 계정만 등록" >>
action.txt
        fi
    detail
    echo ""
    echo "[진단결과]"
    result
    echo ""
    echo "[조치사항]"
    action
}U-18()
{echo ""
echo "[분류]"
echo "파일 및 디렉터리 관리"
echo ""
echo "[항목코드]"

```

```

echo "U-18"
echo ""
echo "[점검항목]"
echo "접속 IP 및 포트 제한"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
####
DD=`cat /etc/hosts.deny`
if [[ "$DD" =~ "ALL: ALL" ]];then
echo "1 /etc/hosts.deny 파일에 ALL Deny 설정이 되어 있음" >> result.txt
else
echo "0 /etc/hosts.deny 파일에 ALL Deny 설정이 되어있지 않음" >> result.txt
echo "/etc/hosts.deny 파일에 ALL Deny 설정 요망" >>action.txt
fi
AA=`cat /etc/hosts.allow`
if [[ "$AA" =~ "sshd" ]];then
echo "1 /etc/hosts.allow 파일 내 허용 ip 설정되어있음" >> result.txt
else
echo "0 /etc/hosts.allow 파일 내 허용 ip 설정되어있지 않음" >> result.txt
echo "/etc/hosts.allow 파일 내 허용 ip 설정요망" >>action.txt
fi
detail
###
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-19()
{echo""
echo [분류]
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-19"
echo ""
echo "[점검항목]"
echo "Finger 서비스 비활성화"

```



```

echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
ls -lL /etc/xinetd.d |grep finger > /dev/null 2>&1
if [ $? -eq 0 ];then
echo "0 Finger 서비스가 존재함" >> result.txt
echo "/ect/xinetd.conf 파일에서 Finger 서비스라인 주석처리 요망" >> action.txt
else
echo "1 Finger 서비스가 존재하지 않음" >> result.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-20()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-20"
echo ""
echo "[점검항목]"
echo "Anonymous FTP 비활성화"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
if [ -f /etc/vsftpd/vsftpd.conf ];then
    if [ "`cat /etc/vsftpd/vsftpd.conf | grep anonymous_enable | awk -F= '{print $2}'" =
NO ];then
        echo "1 익명 ftp 접속 불가능함" >> result.txt
    else
        echo "0 익명 ftp 접속 가능함" >> result.txt
        echo "익명 ftp를 사용하지 않는 경우 차단 설정 적용" >> action.txt
    fi
}

```

```

else
    echo "2 ftp 설치 되어있지 않음" >> result.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-21()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-21"
echo ""
echo "[점검항목]"
echo "r 계열 서비스 비활성화"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
if test -f /etc/xinetd.d/rlogin
    then
        if [ "`cat /etc/xinetd.d/rlogin | grep disable | awk '{print $3}'" = yes ]
            then
                echo "0 rlogin 서비스 활성화 되어있음 " >> result.txt
                echo "rlogin 서비스 비활성화 필요" >> action.txt
            else
                echo "1 rlogin 서비스 비활성화 되어있음" >> result.txt
            fi
        fi
    else
        echo "2 rlogin 설치안되어있음" >> result.txt
    fi
fi
detail
echo ""
echo "[진단결과]"
result
echo ""

```

```

echo "[조치사항]"
action
}U-22()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-22"
echo ""
echo "[점검항목]"
echo "crond 파일 소유자 및 권한 설정"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
ls -al /usr/bin/crontab >/dev/null 2>&1
if [ $? -eq 0 ]; then
    ACP=` find /usr/bin/crontab -perm -750 -ls | grep -v rwsr-x--- | awk '{print $3}'`
    if [ $? -eq 0 ]; then
        if [ $ACP -le 750 ] 2>$ERROR_PATH/error_22; then
            echo 1 crontab 명령어 권한이 750 이하임 >> result.txt
            ADU=` ls -l /usr/bin/crontab | awk '{print $1}'`
            if [ $ADU = 'root' ] 2>$ERROR_PATH/error_22; then
                echo 1 crontab 명령어 소유자가 root임 >> result.txt
            else
                echo 0 crontab 명령어 소유자가 root가 아님 >> result.txt
                echo crontab 명령어 소유자를 root로 변경 요함 >>
action.txt
            fi
        else
            echo 0 crontab 명령어 권한이 750 이하가 아님 >> result.txt
            echo crontab 명령어 권한을 750 이하로 설정 요함 >> action.txt
            if [ $ADU = 'root' ] 2>$ERROR_PATH/error_22; then
                echo 1 crontab 명령어 소유자가 root임 >> result.txt
            else
                echo 0 crontab 명령어 소유자가 root가 아님 >> result.txt
                echo crontab 명령어 소유자를 root로 변경 요함 >>
action.txt
            fi
        fi
    fi
fi

```

```

        fi
else
    echo 2 cron 시스템을 사용하고 있지 않음 >> result.txt
fi
ls -l /etc/cron.allow >/dev/null 2>&1
if [ $? -eq 0 ]; then
    ACP=` find /etc/cron.allow f -type f -perm -640 -ls | grep -v rw-r--r-- | awk '{print $3}'`
    if [ $? -eq 0 ]; then
        if [ $ACP -le 640 ] 2>$ERROR_PATH/error_65; then
            echo 1 cron.allow 파일 권한이 640 이하임 >> result.txt
            ADU=` ls -l /etc/cron.allow | awk '{print $1}'`
            if [ $ADU = 'root' ] 2>$ERROR_PATH/error_65; then
                echo 1 cron.allow 파일 명령어 소유자가 root임
                >> result.txt
            else
                echo 0 cron.allow 파일 명령어 소유자가 root가 아
                님 >> result.txt
                echo cron.allow 파일 명령어 소유자를 root로 변경 요함 >>
                action.txt
            fi
        else
            echo 0 cron.allow 파일 권한이 640 이하가 아님 >> result.txt
            echo cron.allow 파일 권한을 640 이하로 설정 요함 >> action.txt
            if [ $ADU = 'root' ] 2>$ERROR_PATH/error_65; then
                echo 1 cron.allow 파일 명령어 소유자가 root임 >>
                result.txt
            else
                echo 0 cron.allow 파일 명령어 소유자가 root가 아님 >>
                result.txt
                echo cron.allow 파일 명령어 소유자를 root로 변경 요함 >>
                action.txt
            fi
        fi
    fi
else
    echo 2 cron.allow 파일이 없음 >> result.txt
fi
ls -l /etc/cron.deny >/dev/null 2>&1
if [ $? -eq 0 ]; then
    BCP=` find /etc/cron.deny -type f -perm -640 -ls | grep -v 'rw-r-----' | awk '{print $3}'`

```

```

        if [ $BCP -le 640 ] 2>$ERROR_PATH/error_65; then
            echo 1 cron.deny 파일 권한이 640 이하임 >> result.txt
            BDU=`ls -l /etc/cron.deny | awk '{print $3}'`
            if [ $BDU = 'root' ] 2>$ERROR_PATH/error_65; then
                echo 1 cron.deny 파일 명령어 소유자가 root임
>> result.txt
            elif [ $BDU != 'root' ] 2>$ERROR_PATH/error_65; then
                echo 0 cron.deny 파일 명령어 소유자가 root가 아
님 >> result.txt
                echo cron.deny 파일 명령어 소유자를 root로 변경 요함 >>
action.txt
            fi
        elif [ $BCP > 640 ] 2>$ERROR_PATH/error_65; then
            echo 0 cron.deny 파일 권한이 640 이하가 아님 >> result.txt
            echo cron.deny 파일 권한을 640 이하로 설정 요함 >> action.txt
            if [ $BDU = 'root' ] 2>$ERROR_PATH/error_65 ; then
                echo 1 cron.deny 파일 명령어 소유자가 root임 >>
result.txt
            else
                echo 0 cron.deny 파일 명령어 소유자가 root가 아님 >>
result.txt
                echo cron.deny 파일 명령어 소유자를 root로 변경 요함 >>
action.txt
            fi
        elif [ $? -ne 0 ]; then
            echo 2 cron.deny 파일이 없음 >> result.txt
        fi
        detail
        echo ""
        echo "[진단결과]"
        result
        echo ""
        echo "[조치사항]"
        action
    }U-23()
    {echo ""
    echo "[분류]"
    echo "서비스 관리"
    echo ""
    echo "[항목코드]"
    echo "U-23"

```

```

echo ""
echo "[점검항목]"
echo "DoS공격에 취약한 서비스 비활성화"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
###취약점 내용 처리
D1=`find /etc/xinetd.d/echo 2>/dev/null | grep disable | grep no`
D2=`cat /etc/xinetd.d/discard 2>/dev/null | grep disable | grep no`
D3=`cat /etc/xinetd.d/daytime 2>/dev/null | grep disable | grep no`
D4=`cat /etc/xinetd.d/chargen 2>/dev/null | grep disable | grep no`
if [[ -z $D1 ]];then
echo "1 echo 서비스가 설치되어 있지 않거나 비활성화 되어 있음" >> result.txt
else
echo "0 echo 서비스가 활성화 되어 있습니다." >> result.txt
echo "/etc/xinetd.d/ 디렉터리 내 echo 파일 disable = yes로 설정 요망" >> action.txt
fi
if [[ -z $D2 ]];then
echo "1 discard 서비스가 설치되어 있지 않거나 비활성화 되어 있음" >> result.txt
else
echo "0 discard 서비스가 활성화 되어 있음" >> result.txt
echo "/etc/xinetd.d/ 디렉터리 내 discard 파일 disable = yes로 설정 요망" >> action.txt
fi
if [[ -z $D3 ]];then
echo "1 daytime 서비스가 설치되어 있지 않거나 비활성화 되어 있음" >> result.txt
else
echo "0 daytime 서비스가 활성화 되어 있음" >> result.txt
echo "/etc/xinetd.d/ 디렉터리 내 daytime 파일 disable = yes로 설정 요망" >> action.txt
fi
if [[ -z $D4 ]];then
echo "1 chargen 서비스가 설치되어 있지 않거나 비활성화 되어 있음" >> result.txt
else
echo "0 서비스가 활성화 되어 있음" >> result.txt
echo "/etc/xinetd.d/ 디렉터리 내 chargen 파일 disable = yes로 설정 요망" >> action.txt
fi
detail
####
echo ""
echo "[진단결과]"
result

```

```

echo ""
echo "[조치사항]"
action
}U-24()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-24"
echo ""
echo "[점검항목]"
echo "NFS 서비스 비활성화"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
####
NN=`ps -ef | egrep "nfs|statd|lockd"` > /dev/null 2>&1
if [ "$NN" ];then
echo "0 NFS 서비스 동작 중" >> result.txt
echo "NFS서비스 데몬 중지후 시동 스크립트 삭제 또는 스크립트 이름 변경 요망" >>
action.txt
else
echo "1 NFS 서비스가 동작 중이지 않음" >> result.txt
fi
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-25()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-25"

```

```

echo ""
echo "[점검항목]"
echo "NFS 접근통제"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
echo "24번 항목에서 NFS를 비활성화 하는 것을 권장하지만 사용해야 할 경우에는 적절한
접>근통제가 필요함" >> action.txt
echo "이 경우 관리자(=root)가 NFS 서비스를 설치하면서 공유 디렉터리를 임의로 지정하기
>때문에 스크립트로 체크가 불가능" >> action.txt
echo "해당 공유 디렉터리의 권한이 적절한지 수동으로 체크 필요" >> action.txt
echo "2 권한 수동 확인 필요" >> result.txt
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-26()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-26"
echo ""
echo "[점검항목]"
echo "automountd 제거"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
ps -ef | grep automount 2>&1 /dev/null
if [ -n $? ];then
    echo '0 automount 서비스 활성화 되어있음.' >> result.txt
    echo '#kill -9 로 서비스 데몬 중지시켜야한다.' >> action.txt
else
    echo '1 automount 서비스 비활성화 되어있음' >> result.txt
}

```



```

fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-27()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-27"
echo ""
echo "[점검항목]"
echo "RPC 서비스 확인"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
echo "2 관리자의 수동 확인 필요" >> result.txt
echo "RPC 서비스 중지하거나, 최신 버전의 패치 확인" >> action.txt
  detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-28()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-28"
echo ""
echo "[점검항목]"
echo "NIS, NIS+ 점검"

```

```

echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
ps -ef | egrep "ypserv|ypbind|ypxfrd|rpc.yppasswdd|rpc.yppupdated" 2>&1 /dev/null
if [ -n $? ];then
    echo '0 NIS 서비스 활성화 되어있음.' >> result.txt
    echo '#kill -9 PID 로 서비스 데몬 중지시켜야한다.' >> action.txt
else
    echo '1 NIS 서비스 비활성화 되어있음' >> result.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-29()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-29"
echo ""
echo "[점검항목]"
echo "tftp, talk 서비스 비활성화"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
TP=`cat /etc/services | grep tftp | sed -n '1p' | awk '{print $1}`
TK=`cat /etc/services | grep talk | sed -n '1p' | awk '{print $1}`
if [ $TP = W#tftp ];then
    if [ $TK = W#talk ];then
        echo "1 tftp, talk 서비스 비활성화 되어있음" >> result.txt
        echo "시스템 운영에 불필요한 서비스 비활성화 해야함" >> action.txt
    else
        echo "0 tftp 서비스-비활성,talk-활성화 되어있음">> result.txt
    fi
fi
}

```

```

        echo "시스템 운영어 불필요한 서비스 비활성화 해야함" >> action.txt
    fi
else
    if [ $TK = W#talk ];then
        echo "0 tftp-활성화, talk-비활성화 되어있음">> result.txt
        echo "시스템 운영어 불필요한 서비스 비활성화 해야함" >> action.txt
    else
        echo "0 tftp, talk서비스 활성화 되어있음">> result.txt
        echo "시스템 운영어 불필요한 서비스 비활성화 해야함" >> action.txt
    fi
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-30()
{echo ""
echo [분류]
echo 서비스 관리
echo ""
echo [항목코드]
echo U-30
echo ""
echo [점검항목]
echo Sendmail 버전 점검
echo ""
echo [중요도]
echo 상
echo ""
echo [점검현황]
SV=` ps -ef | grep sendmail | grep -v "grep" | wc -l`
if [ $SV -eq 0 ];then
    echo "2 sendmail 서비스를 사용중이지 않음" >> result.txt
else
    echo "2 설치된 sendmail의 버전확인 필요" >> result.txt
echo "최신 버전의 설치 및 업그레이드를 위해 sendmail 데몬의 중지 필요하기 때문에 적절한 시간대에 수행해야 함" >> action.txt
fi
detail

```

```

echo ""
echo [진단결과]
result
echo ""
echo [조치사항]
action
}U-31()
{echo ""
echo "[분류]"
echo "서비스관리"
echo ""
echo "[항목코드]"
echo "U-31"
echo ""
echo "[점검항목]"
echo "스팸 메일 릴레이 제한"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
#####
ps -ef | grep sendmail | grep -v "grep" >/dev/null 2>&1
if [ $? -eq 0 ]; then
    cat $FILE | grep "R$W*" | grep "Relaying denied" | grep -v '#' >/dev/null 2>&1
    if [ $? -eq 0 ]; then
        echo 1 SMTP 릴레이 제한이 설정되어 있음 >> result.txt
    else
        echo 0 SMTP 서비스를 사용하며 릴레이 제한이 설정되어 있지 않음 >>
result.txt
        echo Sendmail 서비스를 사용하지 않을 경우 서비스 중지, 사용할 경우 릴
레이 방지 설정 또는 릴레이 대상 접근 제어 요함 >> action.txt
    fi
else
    echo 1 SMTP 서비스를 사용하지 않음 >> result.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"

```

```

action
}U-32()
{echo ""
echo "[분류]"
echo "서비스관리"
echo ""
echo "[항목코드]"
echo "U-32"
echo ""
echo "[점검항목]"
echo "일반사용자의 Sendmail 실행 방지"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
#####
ps -ef | grep sendmail | grep -v "grep" > /dev/null 2>&1
if [ $? -eq 0 ]; then
    grep -v '^ *#' $FILE | grep PrivacyOptions | grep -v '#' > /dev/null 2>&1
    if [ $? -eq 0 ]; then
        echo 1 일반 사용자의 Sendmail 실행 방지가 설정되어 있음 >>
result.txt
    else
        echo 0 일반 사용자의 Sendmail 실행 방지가 설정되어 있지 않음
>> result.txt
        echo Sendmail 서비스를 사용하지 않을 경우 서비스 중지, Sendmail
서비스를 사용 시 sendmail.cf 설정파일에 restrictqrun 옵션 추가 >> action.txt
    fi
else
    echo 1 SMTP 서비스를 사용하지 않음 >> result.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-33()
{echo ""
echo "[분류]"

```

```

echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-33"
echo ""
echo "[점검항목]"
echo "DNS 보안 버전 패치"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
####
##에러 처리
###취약점 내용 처리
DNS=`ps -ef | grep named`
if [ "$DNS" ];then
echo "0 DNS 서비스 사용중" >> result.txt
echo "DNS 서비스를 사용하지 않는 경우 서비스 중지 요망" >> action.txt
else
echo "1 DNS 서비스 미사용중" >> result.txt
fi
##
N=`named -v 2> $ERROR_PATH/error_33`
if [ -z "$N" ];then
echo "2 보안버전 확인 후 업데이트" >> result.txt
echo "BIND 최신버전 다운로드 사이트 접속 요망" >> action.txt
else
error=`cat $ERROR_PATH/error_33`
echo "2 에러 메시지: "${error}"
fi
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-34()
{echo ""
echo "[분류]"

```

```

echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-34"
echo ""
echo "[점검항목]"
echo "DNS Zone Transfer 설정"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
###취약점 내용 처리
DNS=`ps -ef | grep named`
if [ "$DNS" ];then
echo "0 DNS 서비스 사용중" >> result.txt
echo "/etc/named.conf 파일의 allow-transfer 및 xfrnets확인 후 존파일 전송을 >허용하고자
하는 IP입력" >>action.txt
else
echo "1 DNS 서비스 미사용중" >> result.txt
fi
###
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-35()
{echo " "
echo "[분류]"
echo "서비스 관리"
echo " "
echo "[항목코드]"
echo "U-35"
echo " "
echo "[점검항목]"
echo "웹 서비스 디렉토리 리스팅 제거"
echo " "
echo "[중요도]"
echo "상"
echo " "
}

```

```

echo "[점검현황]"
#####기능#####
IFS_backup="$IFS"
IFS=$'\n'
INDEXES_CHECK=`sed -n '/Options/p' /etc/httpd/conf/httpd.conf >/dev/null 2>&1 | grep -v
"#"`
if [ $? -eq 1 ];then
    echo "2 아파치 설정 파일 확인 불가" >> result.txt
else
for line in ${INDEXES_CHECK}
do
if [ $? -eq 0 ];then
    if [[ ${line} =~ "Indexes" ]];then
        echo "0 Options에 Indexes가 지정되어 있음" >>result.txt
        echo "모든 Options에서 Indexes설정을 제거" >>action.txt
    else
        echo "1 Options에 Indexes가 지정되어 있지 않음" >>result.txt
    fi
else
    error=`cat $ERROR_PATH/error_35`
    echo "2 에러 메시지: "${error}"" >>result.txt
fi
done
fi
IFS="$IFS_backup"
detail
#####
echo " "
echo "[진단결과]"
result
echo " "
echo "[조치사항]"
action
echo " "
}U-36()
{echo " "
echo "[분류]"
echo "서비스 관리"
echo " "
echo "[항목코드]"
echo "U-36"
echo " "
}

```



```

echo "[점검항목]"
echo "웹서비스 웹 프로세스 권한 제한"
echo " "
echo "[중요도]"
echo "상"
echo " "
echo "[점검현황]"
#####기능#####
IFS_backup="$IFS"
IFS=$'\n'
PERMISSIONS_CHECK=`egrep -i "(User|Group)" /etc/httpd/conf/httpd.conf > /dev/null
2>$ERROR_PATH/error_36 |grep -v "#" | grep -v LogFormat`
if [ $? -eq 1 ];then
    echo "2 아파치 설정파일 확인 불가" >> result.txt
else
for name in ${PERMISSIONS_CHECK}
do
if [ $? -eq 0 ];then
    if [ ${name#* } == 'root' ];then
        echo "0 ${name%% *} 권한이 root임." >>result.txt
        echo "웹서비스의 실행 계정을 별도의 계정으로 변경" >>action.txt
    else
        echo "1 ${name%% *} 권한이 root가 아님." >>result.txt
    fi
else
    error=`cat $ERROR_PATH/error_36`
    echo "2 에러 메시지: "${error}"" >>result.txt
    echo "점검할 파일이 존재하지 않음" >>result.txt
fi
done
fi
IFS="$IFS_backup"
detail
#####
echo " "
echo "[진단결과]"
result
echo " "
echo "[조치사항]"
action
}U-37()
{echo " "

```

```

echo "[분류]"
echo "서비스 관리"
echo " "
echo "[항목코드]"
echo "U-37"
echo " "
echo "[점검항목]"
echo "웹서비스 상위 디렉토리 접근금지"
echo " "
echo "[중요도]"
echo "상"
echo " "
echo "[점검현황]"
#####기능#####
IFS_backup="$IFS"
IFS=$'\n'
ALLOWOVERRIDE_CHECK=`sed -n '/AllowOverride/p' /etc/httpd/conf/httpd.conf 2>/dev/null
2>$ERROR_PATH/error_37 | grep -v "#"`
touch result.txt
if [ $? -eq 1 ];then
    echo "2 아파치 설정 파일 확인 불가" >> result.txt
else
for line in ${ALLOWOVERRIDE_CHECK}
do
if [ $? -eq 0 ];then
    if [[ ${line} =~ "None" ]][[ ${line} =~ "none" ]];then
        echo "0 AllowOverride 옵션이 None으로 지정되어 있음" >>result.txt
        echo "모든 AllowOverride 옵션에서 None제거 " >>action.txt
    else
        echo "1 AllowOverride 옵션이 None으로 지정되어 있지 않음" >>result.txt
    fi
else
    error=`cat $ERROR_PATH/error_37`
    echo "2 에러 메시지: "${error}" " >>result.txt
fi
done
fi
IFS="$IFS_backup"
detail
#####
echo " "
echo "[진단결과]"

```

```

result
echo " "
echo "[조치사항]"
action
}U-38()
{echo " "
echo "[분류]"
echo "서비스 관리"
echo " "
echo "[항목코드]"
echo "U-38"
echo " "
echo "[점검항목]"
echo "웹서비스 불필요한 파일 제거"
echo " "
echo "[중요도]"
echo "상"
echo " "
echo "[점검현황]"
if [ `ps -ef | egrep httpd | grep -v "grep" | wc -l` -eq 0 ];then
    echo "2 아파치 미실행 중" >> result.txt
else
    if [ `ls -l /etc/httpd | egrep -i 'manual|samples|docs' | wc -l` -eq 0 ];then
        echo "0 해당항목 설정이 없음" >> result.txt
    fi
fi
detail
echo " "
echo "[진단결과]"
result
echo " "
echo "[조치사항]"
action
}U-39()
{echo " "
echo "[분류]"
echo "서비스 관리"
echo " "
echo "[항목코드]"
echo "U-39"
echo " "
echo "[점검항목]"

```

```

echo "Apache 링크 사용금지"
echo " "
echo "[중요도]"
echo "상"
echo " "
echo "[점검현황]"
#####기능#####
Link_CHECK=`sed -n '/Options/p' /etc/httpd/conf/httpd.conf 2>/dev/null 2>&1 | grep -v "#"`
if [ $? -eq 0 ];then
    if [[ ${Link_CHECK} =~ "FollowSymLinks" ]];then
        echo "0 Options에 FollowSymLinks가 지정되어 있음" >>result.txt
        echo "모든 Options에서 FollowSymLinks설정을 제거 하여 심볼릭 링크 사용
을 제한" >>action.txt
    else
        echo "1 Options FollowSymLinks가 지정되어 있지 않음" >>result.txt
    fi
else
    error=`cat $ERROR_PATH/error_39`
    echo "2 에러 메시지: "${error}" >>result.txt
fi
detail
#####
echo " "
echo "[진단결과]"
result
echo " "
echo "[조치사항]"
action
}U-40()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-40"
echo ""
echo "[점검항목]"
echo "Apache 파일 업로드 및 다운로드 제한"
echo ""
echo "[중요도]"
echo "상"
echo ""

```

```

echo "[점검현황]"
###취약점 내용 처리
AP=`cat /etc/httpd/conf/httpd.conf > /dev/null 2>&1`
if [ -f $AP ];
then
    if [ -z ` $AP | grep "LimitRequestBody" | wc -l ` ];
    then
        echo "1 파일 업로드 및 다운로드를 제한함" >> result.txt
    else
        echo "0 파일 업로드 및 다운로드를 제한하지 않음" >> result.txt
        echo "LimitRequestBody 지시자에서 파일 사이즈 용량 제한" >> action.txt
    fi
else
    error=`cat $ERROR_PATH/error_40`
    echo "2 에러 메시지: "${error}"
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-41()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-41"
echo ""
echo "[점검항목]"
echo "Apache 웹 서비스 영역의 분리"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
###취약점 내용 처리
AP=`cat /etc/httpd/conf/httpd.conf > /dev/null 2>&1`
if [ -f $AP ];
then

```

```

if [ -n ` $AP | grep DocumentRoot | grep -v '#' | grep /var/www/html | wc -l ` ];
then
    echo "0 DocumentRoot가 /var/www/html임" >> result.txt
    echo "별도 디렉터리로 변경" >> action.txt
else
    echo "1 DocumentRoot를 기본 디렉터리로 지정함" >> result.txt
fi
else
error=`cat $ERROR_PATH/error_41`
echo "2 에러 메시지: "${error}"
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-42()
{echo ""
echo [분류]
echo "패치 관리"
echo ""
echo "[항목코드]"
echo "U-42"
echo ""
echo "[점검항목]"
echo "최신 보안패치 및 벤더 권고사항 적용"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
####
###취약점 내용 처리
echo "2 Linux는 서버에 설치된 패치 리스트의 관리가 불가능하므로 rpm 패키지 별 버그가
Fix된 최신 버전 설치가 필요함" >> result.txt
echo -e "<Red Hat 일 경우>\n Step1. http://www.redhat.com/security/updates/에서 해당 버
전 찾을 \n Step2. 현재 사용 중인 보안 관련 Update 찾아 Update Download\n Step3.
Update 설치" >> action.txt
detail
####

```

```

echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-43()
{echo ""
echo "[분류]"
echo "로그 관리"
echo ""
echo "[항목코드]"
echo "U-43"
echo ""
echo "[점검항목]"
echo "로그의 정기적 검토 및 보고"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
####
###취약점 내용 처리
echo "2 로그 기록 검토 및 분석을 시행하여 리포트를 작성하고 정기적으로 보고 해야함" >>
result.txt
echo -e "정기적인 로그 분석을 위하여 아래와 같은 절차 수립\n 1. 정기적인 로그 검토 및
분석 주기 수립\n 2. 로그 분석에 대한 결과 보고서 작성\n 3. 로그 분석 결과보고서 보고 체
계 수립" >> action.txt
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-44()
{echo ""
echo "[분류]"
echo "계정관리"
echo ""
echo "[항목코드]"

```

```

echo "U-44"
echo ""
echo "[점검항목]"
echo "root 이외의 UID가 0 금지"
echo ""
echo "[중요도]"
echo "중"
echo ""
echo "[점검현황]"
####
uid()
{passwd_file=`cut -d: -f1 /etc/passwd`
for name in ${passwd_file}
do
    uid_chk=`grep $name /etc/passwd | grep -v '^root' | cut -d: -f3`
    if [ "$uid_chk" == 0 ];then
        echo "$name 의 UID가 0임"
    fi
done
}uid > 44.txt
if [ -s 44.txt ];then
    echo "0 root 이외의 UID가 0인 계정이 존재함" >> result.txt
    echo "<root 외의 UID가 0인 계정 목록>" >> action.txt
    user_list=`cat 44.txt | awk -F, {print NR "" $0;}`
    echo "${user_list}" >> action.txt
    echo "-----" >> action.txt
    echo "위의 계정에 UID값을 다른 값으로 변경" >> action.txt
else
    echo "1 root 이외의 UID가 0인 계정이 존재하지 않음" >> result.txt
fi
rm -rf 44.txt > /dev/null
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-45()
{echo ""
echo "[분류]"

```



```

echo "계정관리"
echo ""
echo "[항목코드]"
echo "U-45"
echo ""
echo "[점검항목]"
echo "root 계정 su 제한"
echo ""
echo "[중요도]"
echo "하"
echo ""
echo "[점검현황]"
####
cat /etc/group | grep wheel > /dev/null
if [ $? -eq 0 ];then
    echo "1 wheel 그룹이 존재함" >> result.txt
else
    echo "0 wheel 그룹이 존재하지 않음" >> result.txt
fi
find /usr/bin/su -perm 4750 > 45.txt
if [ -s 45.txt ];then
    echo "1 /usr/bin/su 파일의 권한이 4750임" >> result.txt
else
    echo "0 /usr/bin/su 파일의 권한이 4750이 아님" >> result.txt
    echo "/usr/bin/su 파일의 권한을 4750으로 변경">>action.txt
fi
rm -rf 45.txt > /dev/null
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-46()
{echo ""
echo "[분류]"
echo "계정관리"
echo ""
echo "[항목코드]"
echo "U-46"

```

```

echo ""
echo "[점검항목]"
echo "패스워드 최소 길이 설정"
echo ""
echo "[중요도]"
echo "중"
echo ""
echo "[점검현황]"
####
pass_min=`grep PASS_MIN_LEN /etc/login.defs 2>$ERROR_PATH/error_46 | grep -v "^#" `
if [ $? -eq 0 ];then
    pass_min_value=`echo ${pass_min} | cut -d " " -f2`
    if [ ${pass_min_value} -ge 8 ];then
        echo "1 패스워드 최소 길이가 8자 이상으로 설정되어 있음" >> result.txt
    else
        echo "0 패스워드 최소 길이가 8자 미만으로 설정되어 있음" >> result.txt
        echo "/etc/login.defs 파일 내 PASS_MIN_LEN 설정을 8이상으로 변경" >>
action.txt
        fi
    else
        error=`cat $ERROR_PATH/error_46`
        echo "2 에러 메시지: "${error}" " >>result.txt
    fi
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-47()
{echo ""
echo "[분류]"
echo "계정관리"
echo ""
echo "[항목코드]"
echo "U-47"
echo ""
echo "[점검항목]"
echo "패스워드 최대 사용기간 설정"
echo ""

```

```

echo "[중요도]"
echo "중"
echo ""
echo "[점검현황]"
####
pass_maxdays=`grep PASS_MAX_DAYS /etc/login.defs 2>$ERROR_PATH/error_47 | grep -v
"^#" `
if [ $? -eq 0 ];then
    pass_maxdays_value=`echo ${pass_maxdays} | cut -d " " -f2`
    if [ ${pass_maxdays_value} -ge 90 ];then
        echo "0 패스워드 최대 사용기간이 90일 이하로 설정되어 있지 않음" >>
result.txt
        echo "/etc/login.defs 파일 내 PASS_MAX_DAYS 설정을 90 이하로 변경" >>
action.txt
    else
        echo "1 패스워드 최대 사용기간이 90일 이하로 설정되어 있음" >>
result.txt
    fi
else
    error=`cat $ERROR_PATH/error_47`
    echo "2 에러 메시지: "${error}" " >>result.txt
fi
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-48()
{echo ""
echo "[분류]"
echo "계정관리"
echo ""
echo "[항목코드]"
echo "U-48"
echo ""
echo "[점검항목]"
echo "패스워드 최소 사용기간 설정"
echo ""
echo "[중요도]"

```

```

echo "중"
echo ""
echo "[점검현황]"
####
pass_mindays=`grep PASS_MIN_DAYS /etc/login.defs 2>$ERROR_PATH/error_48 | grep -v
"^#" `
if [ $? -eq 0 ];then
    pass_mindays_value=`echo ${pass_mindays} | cut -d " " -f2`
    if [ ${pass_mindays_value} -le 1 ];then
        echo "0 패스워드 최소 사용기간이 설정되어 있지 않음" >> result.txt
        echo "/etc/login.defs 파일 내 PASS_MIN_DAYS 설정을 1 이상으로 변경" >>
action.txt
    else
        echo "1 패스워드 최소 사용기간이 1일 이상으로 설정되어 있음" >>
result.txt
    fi
else
    error=`echo $ERROR_PATH/error_48`
    echo "2 에러 메시지: "${error}" >> result.txt
fi
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-49()
{echo ""
echo "[분류]"
echo "계정관리"
echo ""
echo "[항목코드]"
echo "U-49"
echo ""
echo "[점검항목]"
echo "불필요한 계정 제거"
echo ""
echo "[중요도]"
echo "하"
echo ""
}

```

```

echo "[점검현황]"
echo
echo
-e
"rootWnbinWndaemonWnadmWnlpWnsyncWnshutdownWnhaltWnmailWnnewsWnuucpWnoperator
WngamesWngopherWnftpWnnobodyWndbusWnvcsaWnrpmWnhaldaemonWnidnetWnnetdumpWn
nscdWnsshdWnrpcWnmailnullWnsmmspWnrpcuserWnfnobodyWnncapWnapacheWnsquidWnweb
alizerWnxfswnntpWngdmWnpegasusWnhttWnnamedWnpvm" > default.txt
cut -f 1 -d ':' /etc/passwd > 49.txt
cat default.txt 49.txt | sort | uniq -d > test.txt
cat default.txt 49.txt | sort | uniq -u > test1.txt
a_code=` cat 49.txt | wc -l`
b_code=` cat test1.txt | wc -l`
if [ -e test.txt ];then
    echo "2 ip 포함 ${a_code}개의 계정의 존재. 수동 확인 필요" >> result.txt
    echo "default 계정 삭제 시 업무 영향도 파악 후 userdel 명령어를 이용해 삭제" >>
action.txt
elif [ -e 49.txt ];then
    echo "2 default 계정 외 ${b_code}개의 계정이 존재. 수동 확인 필요" >> result.txt
    echo "불필요한 계정일 시 userdel 명령어를 이용해 삭제" >> action.txt
else
    echo "1 불필요한 계정이 존재하지 않음." >> result.txt
fi
rm default.txt 49.txt test.txt test1.txt
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-50()
{echo ""
echo "[분류]"
echo "계정관리"
echo ""
echo "[항목코드]"
echo "U-50"
echo ""
echo "[점검항목]"
echo "관리자 그룹에 최소한의 계정 포함"
echo ""
echo "[중요도]"
echo "하"

```

```

echo ""
echo "[점검현황]"
####
group_file=`grep -E '^(root|bin|daemon|sys|adm|tty|disk|mem|kmem|wheel):' /etc/group`
for line in `echo ${group_file}`
do
group_name=`echo ${line} | cut -d: -f1`
user=`echo ${line} | cut -d: -f4`
if [ -z ${user} ];then
    echo "1 ${group_name}그룹에 불필요한 계정이 등록되어 있지 않음" >> result.txt
else
    echo "2 (수동확인) ${group_name}그룹에 ${user}계정이 포함되어 있음 " >>
result.txt
    echo "${user} 계정이 반드시 필요한지 확인 필요" >> action.txt
    echo "불필요한 계정일 경우 삭제해야 함" >> action.txt
fi
done
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-51()
{echo ""
echo "[분류]"
echo "계정관리"
echo ""
echo "[항목코드]"
echo "U-51"
echo ""
echo "[점검항목]"
echo "존재하지 않는 GID 금지"
echo ""
echo "[중요도]"
echo "하"
echo ""
echo "[점검현황]"
####
cat /etc/group > group_file

```

```

for group in `cat /etc/group | awk -F: '{print $4}' `
do
    if [ ! -z "$group" ];then
        sed -i "/:${group}/d" group_file
    fi
done
awk -F: '{print $4}' /etc/passwd > pwd_gid
for line in `cat group_file`
do
    num=`echo $line | awk -F: '{print $3}`
    if grep -wq $num pwd_gid ;then
        :
    else
        echo "$line" >> 51.txt
    fi
done
list_51=`awk -F: '{print $1}' 51.txt`
if [ -s 51.txt ];then
    echo "0 존재하지 않는 계정에 GID 설정이 되어 있음" >> result.txt
    echo "<검색된 그룹 목록>" >> action.txt
    echo " 불필요한 그룹이 있을 경우 /etc/group 파일을 검토하여 제거" >> action.txt
    echo "${list_51}">> action.txt
else
    echo "1 존재하지 않는 계정에 GID 설정이 되어 있지 않음" >> result.txt
fi
rm -rf 51.txt >/dev/null 2>&1
rm -rf group_file > /dev/null 2>&1
rm -rf pwd_gid > /dev/null 2>&1
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-52()
{echo ""
echo "[분류]"
echo "계정관리"
echo ""
echo "[항목코드]"

```

```

echo "U-52"
echo ""
echo "[점검항목]"
echo "동일한 UID 금지"
echo ""
echo "[중요도]"
echo "중"
echo ""
echo "[점검현황]"
compare()
{user_uid=`cut -d: -f3 /etc/passwd`
for name in ${user_uid}
do
i=0
    for name2 in ${user_uid}
    do
        if [ ${name} == ${name2} ];then
            let i+=1
        fi
        if [ $i -gt 1 ];then
            echo "${name},${name2}"
            i=0
        fi
    done
done
}compare > 52.txt
if [ -s 52.txt ];then
    echo "0 동일한 UID로 설정된 사용자 계정이 존재함" >> result.txt
    echo "사용자간 다른 UID로 변경" >> action.txt
else
    echo "1 동일한 UID로 설정된 사용자 계정이 존재하지 않음" >> result.txt
fi
rm -rf 52.txt
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-53()

```



```

{echo ""
echo "[분류]"
echo "계정관리"
echo ""
echo "[항목코드]"
echo "U-53"
echo ""
echo "[점검항목]"
echo "사용자 Shell 점검"
echo ""
echo "[중요도]"
echo "하"
echo ""
echo "[점검현황]"
####
shell_chk=`cat /etc/passwd | egrep
"^daemon|^bin|^sys|^listen|^nobody|^nobody4|^noaccess|^diag|^operator|^games|^gopher"
| grep -v "admin"`
for line in `echo "${shell_chk}" | cut -d: -f7`
do
    if [ ${line} == "/sbin/nologin" ];then
        echo "1 로그인 필요하지 않은 계정에 /sbin/nologin 쉘이 부여되지 않음"
>>result.txt
    else
        echo "0 로그인 필요하지 않은 계정에 /sbin/nologin 쉘이 부여되지 않음"
>>result.txt
        echo "로그인이 필요하지 않은 계정에 /sbin/nologin 쉘 부여" >>action.txt
    fi
done
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-54()
{echo ""
echo "[분류]"
echo "계정관리"

```

```

echo ""
echo "[항목코드]"
echo "U-54"
echo ""
echo "[점검항목]"
echo "Session Timeout 설정"
echo ""
echo "[중요도]"
echo "하"
echo ""
echo "[점검현황]"
ST=`cat /etc/profile | grep -v "^#" | grep "^TMOUT"`
if [ $? -eq 0 ];then
    if [ ${ST*=} -le 600 ];then
        echo "1 Session Timeout설정이 600임" >> result.txt
    else
        echo "0 Session Timeout설정이 600이 아님" >> result.txt
        echo "/etc/profile에 Session Timeout을 600으로 설정" >> action.txt
    fi
else
    echo "0 Session Timeout 설정이 되어있지 않음" >> result.txt
    echo "/etc/profile에 Session Timeout을 600으로 설정" >> action.txt
fi
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-55()
{echo ""
echo "[분류]"
echo "파일 및 디렉터리 관리"
echo ""
echo "[항목코드]"
echo "U-55"
echo ""
echo "[점검항목]"
echo "hosts.lpd 파일 소유자 및 권한 설정"
echo ""

```

```

echo "[중요도]"
echo "하"
echo ""
echo "[점검현황]"
hosts=/etc/hosts.lpd
ls -l | grep /etc/hosts.lpd > 55.txt
if [ -s 55.txt ];then
    hosts_check=`ls -l $hosts | awk '{print $3}'`
    if [ ${hosts_check}=='root' ];then
        echo "1 $hosts의 소유자는 root 임" >> result.txt
    else
        echo "0 $hosts의 소유자는 root가 아님" >> result.txt
        echo "$hosts의 소유자를 root로 변경" >> action.txt
    fi
    find $hosts -perm 600 > 55-1.txt
    if [ -s 55-1.txt ];then
        echo "1 $hosts의 권한이 600임" >>result.txt
    else
        echo "$hosts의 권한이 600이 아님" >> result.txt
        echo "$hosts의 권한을 600으로 변경" >> action.txt
    rm -rf 55-1.txt
    fi
else
    echo "2 $hosts파일이 존재하지 않음" >> result.txt
fi
detail
rm -rf 55.txt
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-56()
{echo ""
echo "[분류]"
echo "파일 및 디렉터리 관리"
echo ""
echo "[항목코드]"
echo "U-56"
echo ""
echo "[점검항목]"

```

```

echo "UMASK 설정 관리"
echo ""
echo "[중요도]"
echo "중"
echo ""
echo "[점검현황]"
chk_umask=`umask`
if [ ${chk_umask} -le 0022 ];then
    echo "1 UMASK값이 022이하로 설정되어 있음" >> result.txt
else
    echo "0 UMASK값이 022이하로 설정되어 있지 않음" >> result.txt
    echo "현재 UMASK값 :${chk_umask} " >> action.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-57()
{echo ""
echo "[분류]"
echo "파일 및 디렉터리 관리"
echo " "
echo "[항목코드]"
echo "U-57"
echo " "
echo "[중요도]"
echo "중"
echo " "
echo "[점검항목]"
echo "홈디렉토리 소유자 및 권한 설정"
echo ""
echo "[점검현황]"
###
home=`cat /etc/passwd |grep 'sh$' |awk -F: '{print $6}`
ls -ld $home | while read line
do
    chek=`echo $line | awk '{print $9}' | awk -F/ '{print $NF}`
    f=`echo $line | awk '{print $3}`
    d=`echo $line | awk '{print $9}`

```

```

if [ "$chek" == "$f" ];then
    echo "1 홈 디렉터리 소유자가 해당 계정임" >> result.txt
else
    echo "0 홈 디렉터리 소유자가 해당 계정이 아님" >> result.txt
    echo "사용자별 홈 디렉터리 소유주를 해당 계정으로 변경" >> action.txt
fi
done
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-58()
{echo ""
echo "[분류]"
echo "파일 및 디렉터리 관리"
echo ""
echo "[항목코드]"
echo "U-58"
echo ""
echo "[점검항목]"
echo "홈 디렉터리로 지정한 디렉터리의 존재 관리"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
cat /etc/passwd |awk -F: '{print$1,$6}' >58.txt
echo "2 58.txt파일 참조하여 점검">> result.txt
echo "홈 디렉터리가 존재하지 않는 계정에 홈 디렉터리 설정, 또는 계정 삭제" >> action.txt
rm -rf 58.txt
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-59()
{echo ""

```

```

echo "[분류]"
echo "파일 및 디렉토리 관리"
echo ""
echo "[항목코드]"
echo "U-59"
echo ""
echo "[점검항목]"
echo "숨겨진 파일 및 디렉토리 검색 및 제거"
echo ""
echo "[중요도]"
echo "하"
echo ""
echo "[점검현황]"
#####
find / -type f -name ".*" > A_CODE.txt 2>$ERROR_PATH/error_59
find / -type d -name ".*" >> A_CODE.txt 2>$ERROR_PATH/error_59
if [ -f A_CODE.txt ]; then
    echo 2 숨겨진 파일 및 디렉토리 존재 >> result.txt
    echo 숨겨진 파일 및 디렉토리 존재 파악 후 불법적이거나 의심스러운 파일 삭제요망
>> action.txt
else
    echo 1 숨겨진 파일 및 디렉터리가 없음 >> result.txt
fi
rm A_CODE.txt
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-60()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-60"
echo ""
echo "[점검항목]"
echo "ssh 원격접속 허용"
echo ""

```

```

echo "[중요도]"
echo "중"
echo ""
echo "[점검현황]"
#####
ps -ef | grep telnet | grep -v grep > /dev/null 2>&1
if [ $? -eq 0 ]; then
    echo 0 telnet 서비스를 사용중임 >> result.txt
    echo 원격접속 시 ssh 서비스 설치 및 사용 권한 >> action.txt
elif [ $? -ne 0 ]; then
    ps -ef | grep sshd | grep -v grep > /dev/null 2>&1
    if [ $? -eq 0 ]; then
        echo 1 ssh 서비스를 사용중임 >> result.txt
    elif [ $? -ne 0 ]; then
        echo 0 ssh 서비스를 사용하고 있지 않음 >> result.txt
        echo 원격접속 시 ssh 서비스 설치 및 사용 권한 >> action.txt
    fi
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-61()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-61"
echo ""
echo "[점검항목]"
echo "ssh 원격접속 허용"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
###취약점 내용 처리
SSH=`ps -ef | grep sshd | grep -v grep`

```

```

if [ -n "$SSH" ];then
echo "0 ssh 서비스 사용중" >> result.txt
echo "안전하지 않은 서비스 사용 중지 요망" >> action.txt
else
echo "1 ssh 서비스 미사용중" >> result.txt
fi
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-62()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-62"
echo ""
echo "[점검항목]"
echo "ftp 서비스 확인"
echo ""
echo "[중요도]"
echo "상"
echo ""
echo "[점검현황]"
###취약점 내용 처리
F1=`ps -ef | grep ftp`
F2=`ps -ef | egrep "vstpd |proftp"`
if [ -n "$F1" ];then
echo "0 FTP서비스 활성화" >> result.txt
echo "/etc/xinetd.conf 파일에서 ftp서비스 라인 주석처리 요망" >> action.txt
else
echo "1 FTP서비스 비활성화" >> result.txt
fi
if [ -n "$F2" ];then
echo "0 vsftpd 와 proftp서비스 활성화" >> result.txt
echo " vsftpd 와 proftp서비스 데몬 중지 요망" >> action.txt
else

```



```

echo "1 vsftpd 와 proftp서비스 비활성화" >> result.txt
fi
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-63()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-63"
echo ""
echo "[점검항목]"
echo "ftp 계정 shell 제한"
echo ""
echo "[중요도]"
echo "중"
echo ""
echo "[점검현황]"
###취약점 내용 처리
FS=`cat /etc/passwd | grep ^ftp | awk -F: '{print $7}`
if [ $FS = '/bin/false' ] ; then
echo "1 ftp 계정에 /bin/false 쉘이 부여되어 있음" >> result.txt
else
echo "0 ftp 계정에 /bin/false 쉘이 부여되어 있지 않음" >> result.txt
echo "ftp 계정에 /bin/false 쉘 부여 요망" >> action.txt
fi
detail
####
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-64()

```

```

{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-64"
echo ""
echo "[점검항목]"
echo "ftpusers 파일 설정(FTP 서비스 root 계정 접근제한)"
echo ""
echo "[중요도]"
echo "중"
echo ""
echo "[점검현황]"
#####
#취약점 내용 처리
A_code=/etc/vsftpd/ftpusers
ps -ef | grep vsftpd | grep -v grep >/dev/null 2>&1
if [ $? -ne 0 ]; then
    echo 1 vsFTP 서비스가 비활성화 되어 있음 >> result.txt
else
    cat $A_code | grep ^root >/dev/null 2>&1
    if [ $? -eq 0 ]; then
        echo 1 root 계정 접속이 차단되어 있음 >> result.txt
    else
        echo 0 root 계정 접속이 차단되지 않음 >> result.txt
        echo $A_code에 root 계정 추가 필요 >> action.txt
    fi
fi
B_code=/etc/proftpd.conf
ps -ef | grep proftpd | grep -v grep > /dev/null 2>&1
if [ $? -ne 0 ]; then
    echo 1 proFTP 서비스가 비활성화 되어 있음 >> result.txt
else
    cat $B_code | grep ^RootLogin on > /dev/null 2>&1
    if [ $? -eq 0 ]; then
        echo 1 root 계정 접속이 차단되어 있음 >> result.txt
    else
        echo 0 root 계정 접속이 차단되지 않음 >> result.txt
        echo $B_code에 root 계정 추가 필요 >> action.txt
    fi
fi

```

```

C_code=/etc/ftpusers
ps -ef | grep ftpd | grep -v grep > /dev/null 2>&1
if [ $? -ne 0 ]; then
    echo 1 FTP 서비스가 비활성화 되어 있음 >> result.txt
else
    cat $C_code | grep ^root >/dev/null 2>&1
    if [ $? -eq 0 ]; then
        echo 1 root 계정 접속이 차단되어 있음 >> result.txt
    else
        echo 0 root 계정 접속이 차단되지 않음 >> result.txt
        echo $C_code에 root 계정 추가 필요 >> action.txt
    fi
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-65()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-65"
echo ""
echo "[점검항목]"
echo "at 서비스 권한 설정"
echo ""
echo "[중요도]"
echo "중"
echo ""
echo "[점검현황]"
#####
ls -l /etc/at.allow >/dev/null 2>&1
if [ $? -eq 0 ]; then
    ACP=` find /etc/at.allow f -type f -perm -640 -ls | grep -v rw-r--r-- | awk '{print $3}'`
    if [ $? -eq 0 ]; then
        if [ $ACP =< 640 ] 2>$ERROR_PATH/error_65; then

```

```

echo 1 at.allow 파일 권한이 640 이하임 >> result.txt
ADU=`ls -l /etc/at.allow | awk '{print $1}'`
if [ $ADU = 'root' ] 2> $ERROR_PATH/error_65; then
    echo 1 at.allow 파일 명령어 소유자가 root임 >>
result.txt
else
    echo 0 at.allow 파일 명령어 소유자가 root가 아님
>> result.txt
    echo at.allow 파일 명령어 소유자를 root로 변경 요함 >>
action.txt
fi
else
    echo 0 at.allow 파일 권한이 640 이하가 아님 >> result.txt
echo at.allow 파일 권한을 640 이하로 설정 요함 >> action.txt
if [ $ADU = 'root' ] 2> $ERROR_PATH/error_65; then
    echo 1 at.allow 파일 명령어 소유자가 root임 >> result.txt
else
    echo 0 at.allow 파일 명령어 소유자가 root가 아님 >>
result.txt
    echo at.allow 파일 명령어 소유자를 root로 변경 요함 >> action.txt
fi
fi
else
    echo 2 at.allow 파일이 없음 >> result.txt
fi
ls -l /etc/at.deny >/dev/null 2>&1
if [ $? -eq 0 ]; then
    BCP=`find /etc/at.deny -type f -perm -640 -ls | grep -v 'rw-r-----' | awk '{print $3}'`
    if [ $BCP =< 640 ] 2> $ERROR_PATH/error_65; then
        echo 1 at.deny 파일 권한이 640 이하임 >> result.txt
        BDU=`ls -l /etc/at.deny | awk '{print $3}'`
        if [ $BDU = 'root' ] 2> $ERROR_PATH/error_65; then
            echo 1 at.deny 파일 명령어 소유자가 root임 >>
result.txt
        elif [ $BDU != 'root' ] 2> $ERROR_PATH/error_65; then
            echo 0 at.deny 파일 명령어 소유자가 root가 아님
>> result.txt
        echo at.deny 파일 명령어 소유자를 root로 변경 요함 >>
action.txt
    fi

```

```

        elif [ $BCP > 640 ] 2>$ERROR_PATH/error_65; then
            echo 0 at.deny 파일 권한이 640 이하가 아님 >> result.txt
        echo at.deny 파일 권한을 640 이하로 설정 요함 >> action.txt
        if [ $BDU = 'root' ] 2>$ERROR_PATH/error_65 ; then
            echo 1 at.deny 파일 명령어 소유자가 root임 >> result.txt
        else
            echo 0 at.deny 파일 명령어 소유자가 root가 아님 >>
result.txt
            echo at.deny 파일 명령어 소유자를 root로 변경 요함 >> action.txt
        fi
    fi
elif [ $? -ne 0 ]; then
    echo 2 at.deny 파일이 없음 >> result.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-66()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-66"
echo ""
echo "[점검항목]"
echo "SNMP 서비스 구동 점검"
echo ""
echo "[중요도]"
echo "중"
echo ""
echo "[점검현황]"
comma=`ps -ef | grep 'snmp' > /dev/null`
if [ $? -eq 0 ];then
    echo "1 SNMP 서비스 실행중이 아님" >> result.txt
else
    echo "0 SNMP 서비스 실행중임" >> result.txt
    echo "'service snmpd stop'으로 서비스 중지 필요" >> action.txt
}

```

```

fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-67()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-67"
echo ""
echo "[점검항목]"
echo "SNMP 서비스 Community String 복잡성 설정"
echo ""
echo "[중요도]"
echo "중"
echo ""
echo "[점검현황]"
COMMAN=`cat /etc/snmp/snmpd.conf >/dev/null 2>&1 | egrep '(public|private)' | grep -v '#`
if [ $? -eq 0 ];then
    echo "1 SNMP Community 이름이 public, private이 아님" >> result.txt
else
    echo "0 SNMP Community 이름이 public, private임" >> result.txt
    echo "SNMP 설정파일을 열어 Community String 값 설정 변경" >> action.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-68()
{echo ""
echo "[분류]"
echo "서비스 관리"

```

```

echo ""
echo "[항목코드]"
echo "U-68"
echo ""
echo "[점검항목]"
echo "로그온 시 경고 메시지 제공"
echo ""
echo "[중요도]"
echo "하"
echo ""
echo "[점검현황]"
FILE=/etc/issue.net
if [ -s $FILE ];then
    echo "1 서버 및 Telnet 서비스에 로그인 메시지가 설정되어있음" >> result.txt
else
    echo "0 서버 및 Telnet 서비스에 로그인 메시지가 설정 되어있지않음" >> result.tx
t    echo "Telenet, FTP, SMTP, DNS 서비스를 사용하는 경우 설정파일 조치 후 inetd 데
문 재시작해야함" >> action.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-69()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-69"
echo ""
echo "[점검항목]"
echo "NFS 설정파일 접근권한"
echo ""
echo "[중요도]"
echo "중"
echo ""
echo "[점검현황]"
RT=`ls -l /etc/exports | awk -F . '{ print $1 }'`

```

```

RT1=`echo ${RT} | cut -c2-4 `
RT2=`echo ${RT} | cut -c5-7 `
RT3=`echo ${RT} | cut -c8-11`
echo "현재 /etc/exports의 권한 ${RT}"
if [ ${RT1} == "rwx" ];then
    echo -e "0 644이하로 설정되어 있지 않음" >> result.txt
    echo "644이하로 설정해야 함" >> action.txt
elif [ ${RT1} == "rw-" ];then
    if [ ${RT2} == "rwx" ] || [ ${RT2} == "rw-" ] || [ ${RT2} == "r-x" ];then
        echo "0 644이하로 설정되어 있지 않음" >> result.txt
    elif [ ${RT3} == "rwx" ] || [ ${RT3} == "rw-" ] || [ ${RT3} == "r-x" ];then
        echo "0 644이하로 설정되어 있지 않음" >> result.txt
    else
        echo "1 644이하로 설정 되어 있음" >> result.txt
    fi
else
    echo "1 644이하로 설정 되어 있음" >> result.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-70()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-70"
echo ""
echo "[점검항목]"
echo "expn, vrfy 명령어 제한"
echo ""
echo "[중요도]"
echo "중"
echo ""
echo "[점검현황]"
SMAIL=`cat /etc/mail/sendmail.cf > /dev/null 2>&1 | grep PrivacyOptions`
if [ $? -eq 0 ];then

```



```

        echo "1 noexpn, novrfy 옵션이 설정 되어있음" >> result.txt
else
        echo "0 noexpn, novrfy 옵션이 설정 안되어있음" >> result.txt
        echo "/etc/mail/sendmail.cf에 authwarnings,novrfy,noexpn 옵션 추가 해야함" >>
action.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
}U-71()
{echo ""
echo "[분류]"
echo "서비스 관리"
echo ""
echo "[항목코드]"
echo "U-71"
echo ""
echo "[점검항목]"
echo "Apache 웹 서비스 정보 숨김"
echo ""
echo "[중요도]"
echo "중"
echo ""
echo "[점검현황]"
ST=`cat /etc/httpd/conf/httpd.conf > /dev/null 2>&1| grep -i servertokens | awk '{print $2}'`
if [ $? == 'Prod' ];then
        echo "1 ServerTokens Prod, ServerSignature Off로 설정되어 있음" >> result.txt
else
        echo "0 ServerTokens Prod, ServerSignature Off로 설정되어 있지않음" >> result.txt
        echo "httpd.conf 파일을 열어 ServerTokens Prod, ServerSignature Off 지시자에 Off
옵션 설정" >> action.txt
fi
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"

```

```

action
}U-72()
{echo ""
echo "[분류]"
echo "로그 관리"
echo ""
echo "[항목코드]"
echo "U-72"
echo ""
echo "[점검항목]"
echo "정책에 따른 시스템 로깅 설정"
echo ""
echo "[중요도]"
echo "하"
echo ""
echo "[점검현황]"
#####
echo 2 관리자의 수동 확인 필요 >> result.txt
echo 로그 기록 정책을 수립하고, 정책에 따라 syslog.conf 파일 설정 요망 >> action.txt
detail
echo ""
echo "[진단결과]"
result
echo ""
echo "[조치사항]"
action
rm -rf 2 1>/dev/null 2>&1
rm -rf 640 >/dev/null 2>&1
}

```

result.sh

```

#!/bin/bash
detail()
{
sort -u result.txt | cut -b3-
}
result()
{
RESULT=`cut -f 1 -d ' ' result.txt`
echo "${RESULT}" | grep "^1" >>result1.txt
}

```

```

echo "${RESULT}" | grep "^0" >>result0.txt
echo "${RESULT}" | grep "^2" >>result2.txt
if [ -s result2.txt ];then
    echo "확인필요"
elif [ -s result1.txt ] && [ -s result0.txt ];then
    echo "부분만족"
elif [ -s result1.txt ];then
    echo "만족"
elif [ -s result0.txt ];then
    echo "불만족"
fi
rm -rf result1.txt >/dev/null
rm -rf result0.txt >/dev/null
rm -rf result2.txt >/dev/null
rm -rf result.txt >/dev/null
}
action()
{
sed '$!N;/^W(. *W)WnW1$/!P;D' action.txt 2>/dev/null
rm -rf action.txt >/dev/null
}

```

### all.sh

```

#!/bin/bash
. function.sh
MAIN_IP=`/sbin/ifconfig | grep 'W<inetW>' | sed -n '1p' |tr -s ' '|cut -d ' ' -f3`
MAIN_PATH=`dirname $0`
MAIN_DATE=`date '+%Y%m%d'`
RESULT_DETAILS="${MAIN_PATH}/${MAIN_IP}_${MAIN_DATE}.txt"
echo "취약점 점검 시작"
for i in $( seq -f "%02g" 1 72 )
do
    U-$i >> ${RESULT_DETAILS}
    case $i in
        01)
            echo -ne "[#                               ]05%Wr"
            sleep 1
            ;;
        05)
            echo -ne "[##                               ]10%Wr"
            sleep 1
            ;;
    esac
done

```

```

10)
    echo -ne "[####                ]20%Wr"
    sleep 1
    ;;
20)
    echo -ne "[#####              ]40%Wr"
    sleep 1
    ;;
40)
    echo -ne "[#####              ]50%Wr"
    sleep 1
    ;;
50)
    echo -ne "[#####              ]70%Wr"
    sleep 1
    ;;
60)
    echo -ne "[#####              ]80%Wr"
    sleep 1
    ;;
70)
    echo -ne "[#####              ]99%Wr"
    sleep 1
    ;;
72)
    echo "[#####              ]100%"
    sleep 1
    ;;

    esac
done
echo "[END]" >> ${RESULT_DETAILS}
echo "취약점 점검 완료!"

```

**account.sh**

```

#!/bin/bash
. function.sh
MAIN_IP=`/sbin/ifconfig | grep 'W<inetW>' | sed -n '1p' |tr -s ' '|cut -d ' ' -f3`
MAIN_PATH=`dirname $0`
MAIN_DATE=`date +%Y%m%d`
RESULT_DETAILS="${MAIN_PATH}/${MAIN_IP}_${MAIN_DATE}.txt"
echo "계정 관리 점검"

```

```

echo -ne "[##                               ]01%Wr"
sleep 1
for i in $( seq -f "%02g" 1 4 )
do
    U-$i >> ${RESULT_DETAILS}
    case $i in
        04)
            echo -ne "[####                               ]20%Wr"
            sleep 1
            ;;
    esac
done
for i in $( seq -f "%02g" 44 54 )
do
    U-$i >> ${RESULT_DETAILS}
    case %i in
        48)
            echo -ne "[#####                               ]50%Wr"
            sleep 1
            ;;
        50)
            echo -ne "[#####                               ]80%Wr"
            sleep 1
            ;;
        54)
            echo -ne "[#####                               ]100%Wr"
            sleep 1
            ;;
    esac
done
echo "계정 관리 점검 완료"

```

### file.sh

```

#!/bin/bash
. function.sh
MAIN_IP=`/sbin/ifconfig | grep 'W<inetW>' | sed -n '1p' |tr -s ' '|cut -d ' ' -f3`
MAIN_PATH=`dirname $0`
MAIN_DATE=`date '+%Y%m%d'`
RESULT_DETAILS="${MAIN_PATH}/${MAIN_IP}_${MAIN_DATE}.txt"
echo "파일 및 디렉터리 관리 점검"
echo -ne "[##                               ]01%Wr"

```

```

sleep 1
for i in $( seq -f "%02g" 5 18 )
do
    U-$i >> ${RESULT_DETAILS}
    case $i in
        06)
            echo -ne "[####                ]20%Wr"
            sleep 1
            ;;
        10)
            echo -ne "[#####              ]50%Wr"
            sleep 1
            ;;
        15)
            echo -ne "[#####              ]80%Wr"
            sleep 1
            ;;
    esac
done
for i in $( seq -f "%02g" 55 59 )
do
    U-$i >> ${RESULT_DETAILS}
    case %i in
        59)
            echo -ne "[#####              ]100%Wr"
            sleep 1
            ;;
    esac
done
echo "파일 및 디렉터리 관리 완료"

```

### service.sh

```

#!/bin/bash
. function.sh
MAIN_IP=`/sbin/ifconfig | grep 'W<inetW>' | sed -n '1p' |tr -s ' '|cut -d ' ' -f3`
MAIN_PATH=`dirname $0`
MAIN_DATE=`date '+%Y%m%d'`
RESULT_DETAILS="${MAIN_PATH}/${MAIN_IP}_${MAIN_DATE}.txt"
echo "서비스 관리 점검"
echo -ne "[##                ]01%Wr"

```

```

sleep 1
for i in $( seq -f "%02g" 19 43 )
do
    U-$i >> ${RESULT_DETAILS}
    case $i in
        30)
            echo -ne "[####                ]20%Wr"
            sleep 1
            ;;
        40)
            echo -ne "[#####              ]50%Wr"
            sleep 1
            ;;
    esac
done
for i in $( seq -f "%02g" 60 72 )
do
    U-$i >> ${RESULT_DETAILS}
    case %i in
        65)
            echo -ne "[#####              ]80%Wr"
            sleep 1
            ;;
        72)
            echo -ne "[#####              ]100%Wr"
            sleep 1
            ;;
    esac
done
echo "서비스 관리 완료"

```

### final\_DB.sh

```

#!/bin/bash
echo -n "FILE NAME: "
read file
echo -n "USER ID: "
read user_id
final_result()
{## hostname
host_name=`hostname -f `
## 전체 항목 : 72

```

```

## 만족 개수
line_com=`grep -n "[진단결과]" $file | cut -d: -f1`
echo "${line_com}" > line.txt
num0=0
num1=0
num01=0
num2=0
for line in `cat line.txt`
do
com_line=`expr ${line} + 1`
fin_line=`sed -n ${com_line}p $file`
if [ ${fin_line} == "만족" ];then
    let num1+=1
elif [ ${fin_line} == "부분만족" ];then
    let num01+=1
elif [ ${fin_line} == "불만족" ];then
    let num0+=1
elif [ ${fin_line} == "확인필요" ];then
    let num2+=1
else
    echo "오류"
fi
done
## 점검항목
line_num=`grep -n "[분류]" $file | cut -d: -f1`
echo "${line_num}" > line_num.txt
num=0
for line in `cat line_num.txt`
do
let num+=1
done
all=`expr ${num} - ${num2}`
### 보안점수 계산
### 만족/전체개수 * 100
#echo "scale=2; $num1 / $num * 100" | bc
#####
echo "insert into result values(null,${user_id},'${host_name}',72,$all,$num1,$num0,$num2,"
echo "scale=2; $num1 / $num * 100" | bc
echo "%"
echo ",now());"
}final_detail()
{line_end_num=`grep -n "END" $file | cut -d: -f1`

```



```

line1_num=`grep -n "W[" $file | cut -d: -f1`
grep -n "W[" $file | cut -d: -f1 > line2_num
echo "${line1_num}" > line1_num.txt
sed -n "2, W$p" line2_num > line2_num.txt
paste -d : line1_num.txt line2_num.txt > com_line.txt
num=0
for line in `cat com_line.txt`
do
let num+=1
line1=`echo ${line} | cut -d: -f1`
line2=`echo ${line} | cut -d: -f2`
if [ ${line1} == ${line_end_num} ];then
echo "end"
break
fi
number1=`expr ${line1} + 1`
number2=`expr ${line2} - 1`
### 명령어 입력
if [ `expr $num % 7` -eq 1 ];then
echo "insert into detail values(null,@result_id,"
fi
echo ""
sed -n "${number1},${number2}p" $file
echo ""
if [ `expr $num % 7` -eq 0 ];then
echo ");"
else
echo ","
fi
done
}#final_detail
final_result > result_final.txt 2>/dev/null
final_detail > detail_ex.txt
cat detail_ex.txt | sed '$d' | tr "\n" " " > detail_final.txt 2>/dev/null
DB_ex()
{#####
cat result_final.txt | tr "\n" " "
echo "set @result_id = last_insert_id();"
cat detail_final.txt
#####
}DB_ex > DB.txt
echo "complete!"

```

```
rm -rf detail_ex.txt
rm -rf detail_final.txt
rm -rf line.txt
rm -rf line1_num.txt
rm -rf line2_num
rm -rf line2_num.txt
rm -rf line_num.txt
rm -rf result_final.txt
rm -rf com_line.txt
```

## 5.2 발표자료

**IR(Inform Relief) System**

- 시스템 취약점 점검 -

JOONGBU UNIVERSITY  
JBU  
중부대학교

팀명: Checker  
팀장: 조서연  
팀원 : 김미란, 김성한, 김예리, 이윤지  
지도 교수 : 양환석 교수님

목차

구상도                      시연

작품 소개                      IR code                      마무리

## 작품 소개

- 웹 페이지에서 SSH접속을 통해 시스템 취약점 점검 자동화 서비스를 제공
- 리눅스 운영체제를 대상으로 점검
- 취약점을 점검하고 나면 결과 파일은 DB를 통해 웹서버로 관리
- 이용자는 웹 서비스를 통해 시스템 취약점 점검의 결과를 열람 및 다운로드가 가능함

## 작품 소개

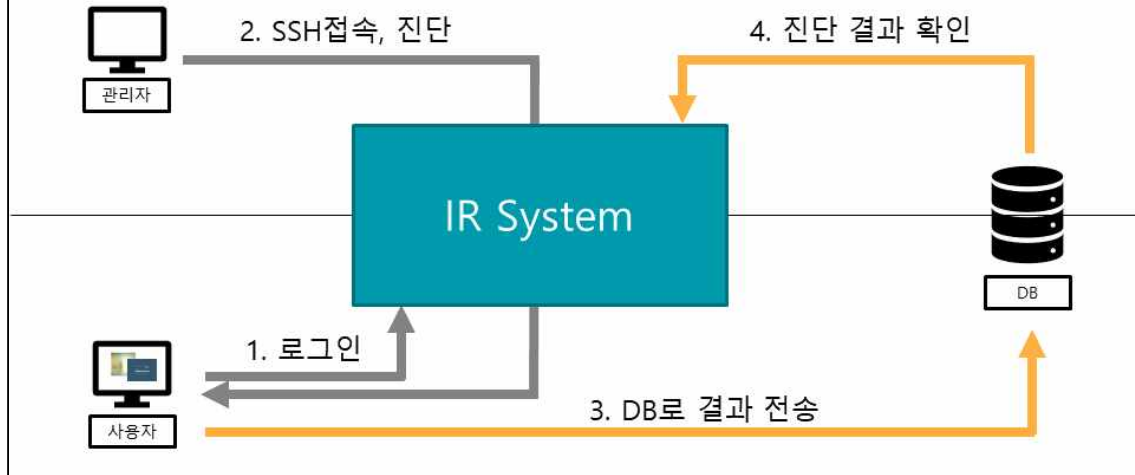
- 2017 '주요정보통신기반시설 기술적 취약점 분석 평가 방법 상세가이드'를 기반으로 함



항목	종류	페이지
SSH 접속 권한 점검	필	14.01
패스워드 복잡도 평가	필	14.02
계정 잠금 임계값 설정	필	14.03
패스워드 기록 보존	필	14.04
SSH 접속 로그 기록	필	14.05
패스워드 최소 길이 설정	필	14.06
패스워드 최소 사용기간 설정	필	14.07
패스워드 최소 사용횟수 제한	필	14.08
불필요한 계정 제거	필	14.09
중요한 계정 비밀번호의 계정 로컬	필	14.10
계정의 존재하지 않는 SSH 키	필	14.11
동일한 SSH 키	필	14.12
이름 없는 SSH 키	필	14.13
Session Timeout 설정	필	14.14
SSH 클라이언트 인증 및 접근 설정	필	14.15
파일 및 디렉토리 소유자 설정	필	14.16
rsyncd.conf 파일 소유자 및 권한 설정	필	14.17
rsyncd.conf 파일 소유자 및 권한 설정	필	14.18
rsyncd.conf 파일 소유자 및 권한 설정	필	14.19
rsyncd.conf 파일 소유자 및 권한 설정	필	14.20
rsyncd.conf 파일 소유자 및 권한 설정	필	14.21
rsyncd.conf 파일 소유자 및 권한 설정	필	14.22
rsyncd.conf 파일 소유자 및 권한 설정	필	14.23
rsyncd.conf 파일 소유자 및 권한 설정	필	14.24
rsyncd.conf 파일 소유자 및 권한 설정	필	14.25
rsyncd.conf 파일 소유자 및 권한 설정	필	14.26
rsyncd.conf 파일 소유자 및 권한 설정	필	14.27
rsyncd.conf 파일 소유자 및 권한 설정	필	14.28
rsyncd.conf 파일 소유자 및 권한 설정	필	14.29
rsyncd.conf 파일 소유자 및 권한 설정	필	14.30
rsyncd.conf 파일 소유자 및 권한 설정	필	14.31
rsyncd.conf 파일 소유자 및 권한 설정	필	14.32
rsyncd.conf 파일 소유자 및 권한 설정	필	14.33
rsyncd.conf 파일 소유자 및 권한 설정	필	14.34
rsyncd.conf 파일 소유자 및 권한 설정	필	14.35
rsyncd.conf 파일 소유자 및 권한 설정	필	14.36
rsyncd.conf 파일 소유자 및 권한 설정	필	14.37
rsyncd.conf 파일 소유자 및 권한 설정	필	14.38
rsyncd.conf 파일 소유자 및 권한 설정	필	14.39
rsyncd.conf 파일 소유자 및 권한 설정	필	14.40

구분	내용
1. 개요	1.1. 목적
2. 적용 대상	2.1. 대상 시스템
3. 평가 방법	3.1. 평가 방법
4. 결과물	4.1. 결과물
5. 참고	5.1. 참고
6. 부록	6.1. 부록
7. 용어	7.1. 용어
8. 약어	8.1. 약어
9. 기타	9.1. 기타
10. 결론	10.1. 결론
11. 감사	11.1. 감사
12. 저작권	12.1. 저작권
13. 라이선스	13.1. 라이선스
14. 연락처	14.1. 연락처
15. 이력	15.1. 이력
16. 기타	16.1. 기타
17. 기타	17.1. 기타
18. 기타	18.1. 기타
19. 기타	19.1. 기타
20. 기타	20.1. 기타
21. 기타	21.1. 기타
22. 기타	22.1. 기타
23. 기타	23.1. 기타
24. 기타	24.1. 기타
25. 기타	25.1. 기타
26. 기타	26.1. 기타
27. 기타	27.1. 기타
28. 기타	28.1. 기타
29. 기타	29.1. 기타
30. 기타	30.1. 기타
31. 기타	31.1. 기타
32. 기타	32.1. 기타
33. 기타	33.1. 기타
34. 기타	34.1. 기타
35. 기타	35.1. 기타
36. 기타	36.1. 기타
37. 기타	37.1. 기타
38. 기타	38.1. 기타
39. 기타	39.1. 기타
40. 기타	40.1. 기타
41. 기타	41.1. 기타
42. 기타	42.1. 기타
43. 기타	43.1. 기타
44. 기타	44.1. 기타
45. 기타	45.1. 기타
46. 기타	46.1. 기타
47. 기타	47.1. 기타
48. 기타	48.1. 기타
49. 기타	49.1. 기타
50. 기타	50.1. 기타
51. 기타	51.1. 기타
52. 기타	52.1. 기타
53. 기타	53.1. 기타
54. 기타	54.1. 기타
55. 기타	55.1. 기타
56. 기타	56.1. 기타
57. 기타	57.1. 기타
58. 기타	58.1. 기타
59. 기타	59.1. 기타
60. 기타	60.1. 기타
61. 기타	61.1. 기타
62. 기타	62.1. 기타
63. 기타	63.1. 기타
64. 기타	64.1. 기타
65. 기타	65.1. 기타
66. 기타	66.1. 기타
67. 기타	67.1. 기타
68. 기타	68.1. 기타
69. 기타	69.1. 기타
70. 기타	70.1. 기타
71. 기타	71.1. 기타
72. 기타	72.1. 기타
73. 기타	73.1. 기타
74. 기타	74.1. 기타
75. 기타	75.1. 기타
76. 기타	76.1. 기타
77. 기타	77.1. 기타
78. 기타	78.1. 기타
79. 기타	79.1. 기타
80. 기타	80.1. 기타
81. 기타	81.1. 기타
82. 기타	82.1. 기타
83. 기타	83.1. 기타
84. 기타	84.1. 기타
85. 기타	85.1. 기타
86. 기타	86.1. 기타
87. 기타	87.1. 기타
88. 기타	88.1. 기타
89. 기타	89.1. 기타
90. 기타	90.1. 기타
91. 기타	91.1. 기타
92. 기타	92.1. 기타
93. 기타	93.1. 기타
94. 기타	94.1. 기타
95. 기타	95.1. 기타
96. 기타	96.1. 기타
97. 기타	97.1. 기타
98. 기타	98.1. 기타
99. 기타	99.1. 기타
100. 기타	100.1. 기타

## 구상도



## IR Code (Linux)

<pre> root@localhost/~# U-35 항목 코드 파살(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H) echo - echo -[분류]- echo -서비스 관리- echo - echo -[항목코드]- echo -U-35- echo - echo -[점검항목]- echo -웹 서비스 디렉토리 리스칭 제거- echo - echo -[중요도]- echo -상- echo - echo -[점검현황]- echo -#####- #####기동##### IFS=\$'\n' IFS_BACKUP=\$IFS IFS=\$'\n' INDEXES_CHECK=\$(cat /dev/null &gt;&gt; /dev/null 2&gt;&amp;1) for line in \$(INDEXES_CHECK); do if [ \$? -eq 0 ]; then if [ [ \$line </pre>	<pre> root@localhost/~# U-35 항목 결과 [점검현황] 웹 서비스 디렉토리 리스칭 제거 [중요도] 상 [점검분류] AllowOverride 옵션이 None으로 지정되어 있음 [점검항목] Options에 Indexes가 지정되어 있음 Options에 Indexes가 지정되어 있지 않음 [진단결과] 부분만족 [조치사항] 모든 Options에서 Indexes옵션을 제거 </pre>	<pre> root@localhost/~# U-39 항목 코드 파살(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H) echo -서비스 관리- echo - echo -[항목코드]- echo -U-39- echo - echo -[점검항목]- echo -Apache 링크 사용금지- echo - echo -[중요도]- echo -상- echo - echo -[점검현황]- echo -#####- #####기동##### IFS=\$'\n' IFS_BACKUP=\$IFS IFS=\$'\n' INDEXES_CHECK=\$(cat /dev/null &gt;&gt; /dev/null 2&gt;&amp;1) for line in \$(INDEXES_CHECK); do if [ \$? -eq 0 ]; then if [ [ \$line </pre>	<pre> root@localhost/~# U-39 항목 결과 [점검현황] AllowOverride 옵션이 None으로 지정되어 있음 [진단결과] 불만족 [조치사항] 모든 AllowOverride 옵션에서 None제거 [분류] 서비스 관리 </pre>
--	--	--	--

## IR System - User



[그림 U-1] 웹 페이지

## IR System - User



[그림 U-2] 회원가입



[그림 U-3] 로그인



## IR System - Admin

The screenshot shows a web application interface with a dark header containing 'HOME' and navigation links. The main content area is divided into three sections:

- 회원 정보**: Displays user information such as '이름 : 조서원', '계정 : test', '생년월일 : 2001-10-12', and '가입 번호 : 2021-10-12 04181544'.
- 인단 결과**: A table with columns for '번호', '인원비율', '점수', '평가', '수정일자', '평가', 'NA', and '비밀번호'. It contains one row of data.
- 상세 결과**: A table with columns for '번호', '평가비율', '점수', '평가', '인원비율', '평가', and '평가비율'. It contains multiple rows of data, each with detailed text descriptions.

[그림 A-8] 점검 결과 확인

## IR System - Admin

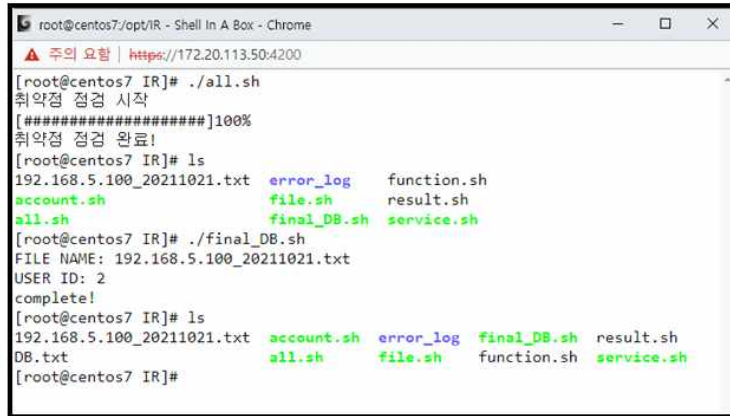
```

root@centos7:~/opt/IR - Shell In A Box - Chrome
주의 요함 | https://172.20.113.50:4200
[root@centos7 IR]# ls
192.168.5.100_20211021.txt  account.sh  error_log  final_DB.sh  result.sh
DB.txt                    all.sh      file.sh     function.sh  service.sh
[root@centos7 IR]# mysql -h 172.20.113.175 -u root -p vulnerability < DB.txt
    
```

[그림 A-7] DB로 전송



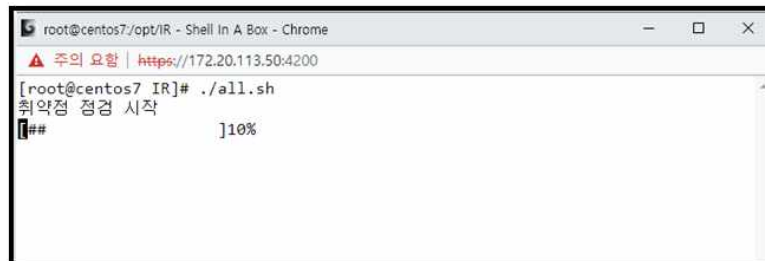
## IR System - Admin



```
root@centos7/opt/IR - Shell In A Box - Chrome
주의 요함 | https://172.20.113.50:4200
[root@centos7 IR]# ./all.sh
취약점 점검 시작
[#####]100%
취약점 점검 완료!
[root@centos7 IR]# ls
192.168.5.100_20211021.txt  error_log  function.sh
account.sh                file.sh    result.sh
all.sh                    final_DB.sh  service.sh
[root@centos7 IR]# ./final_DB.sh
FILE NAME: 192.168.5.100_20211021.txt
USER ID: 2
complete!
[root@centos7 IR]# ls
192.168.5.100_20211021.txt  account.sh  error_log  final_DB.sh  result.sh
DB.txt                     all.sh      file.sh    function.sh   service.sh
[root@centos7 IR]#
```

[그림 A-6] 점검 완료 후 DB 가공

## IR System - Admin



```
root@centos7/opt/IR - Shell In A Box - Chrome
주의 요함 | https://172.20.113.50:4200
[root@centos7 IR]# ./all.sh
취약점 점검 시작
[##]10%
```

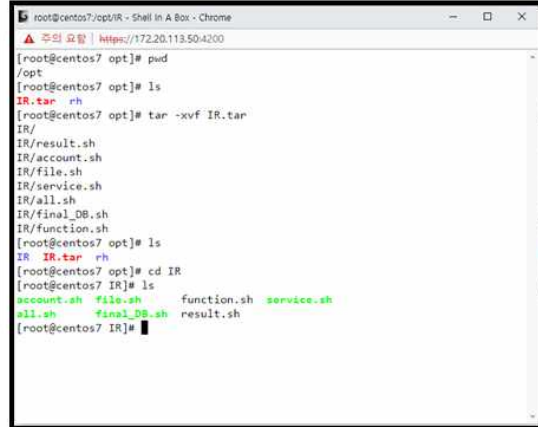
[그림 A-5] 진단 시작

## IR System - Admin



```
C:\Users\김성환>scp C:\Users\김성환\Desktop\IR.tar root@172.20.113.50:/opt
```

[그림 A-3] IR Code 전송



```
root@centos7/opt/IR - Shell in A Box - Chrome
[root@centos7 opt]# pwd
/opt
[root@centos7 opt]# ls
IR
[root@centos7 opt]# tar -xvf IR.tar
IR/
IR/result.sh
IR/account.sh
IR/file.sh
IR/service.sh
IR/all.sh
IR/final_DB.sh
IR/function.sh
[root@centos7 opt]# ls
IR IR.tar
[root@centos7 opt]# cd IR
[root@centos7 IR]# ls
account.sh file.sh function.sh service.sh
all.sh final_DB.sh result.sh
[root@centos7 IR]#
```

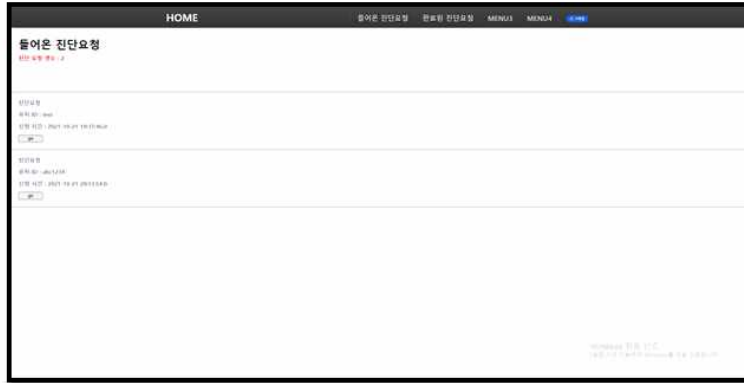
[그림 A-4] 해당 시스템 접속

## IR System - Admin



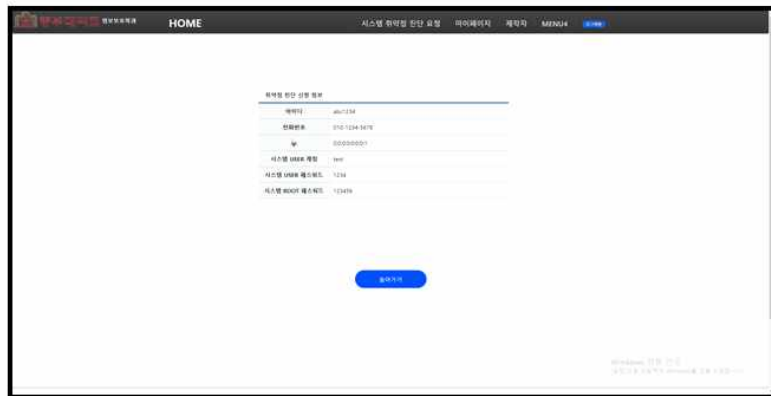
[그림 A-2] 특정 시스템 요청 확인

## IR System - Admin



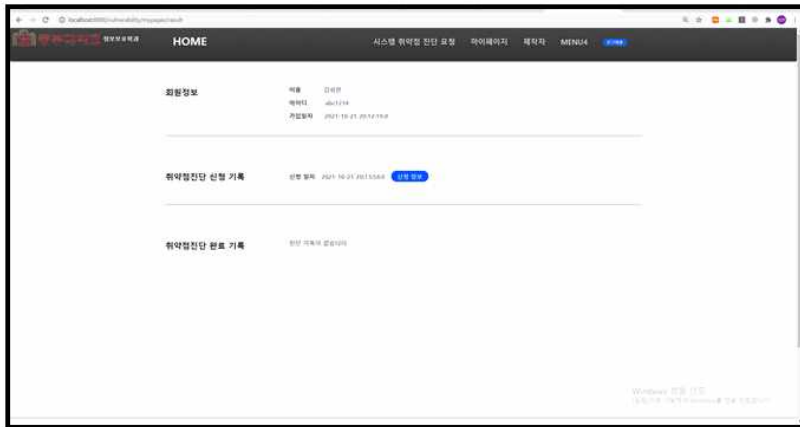
[그림 A-1] 요청 리스트

## IR System - User



[그림 U-6] 사용자의 신청 정보 확인

## IR System - User



[그림 U-5] 사용자의 마이페이지

