Telegram-bot을 활용한 서버 취약점 진단

팀 명 : 일석이조

지도 교수 : 양환석 교수님

팀 장 : 남승택

팀 원 : 신자연

박세빈

왕나원

양재희

2021. 11.

중부대학교 정보보호학과

목 차

5. 별첨

1. 서론

1.1 연구 배경

서버가 없는 프로그램이 없을 정도로 많은 사람이 서버를 이용하고 있다. 이와 함께 서버의 취약함을 이용한 위협요소가 급증하고 있는 추세다. 우리는 이 현상에 주목해 서버의 취약점을 미리 진단하고 예방하는 효과적인 방법을 탐구하기로 하였다.

1.2 연구 필요성

서버는 주된 정보의 제공이나 작업을 수행하는 컴퓨터 시스템이다. 이러한 서버의 취약점을 이용하여 공격하는 사례가 늘어나고 있다. 2021년은 코로나로 인해 재택근무 같은 원격 액세스 인터페이스를 교두보로 삼는 공급만 공격이 증가한 만큼, 이를 타깃으로 한 서버 보안 문제가 코로나19가 있기 전보다 10% 이상 증가했다[1]. 이로 인해 정보통신 기반 보호법 제9조에 따라 관리기관은 매년 취약점 분석 평가를 실시해야 한다. 그러나 대부분의 관리자가 자신의 서버가 어느 부분이 취약하고 보완해야 할 점이 어떤 부분인지 정확히 알지 못하기 때문에 '취약점 진단'이 필요하다. 취약점 진단은 필수적으로 필요하며 사소해 보이는 취약점을 찾아서 위험 요소에 대한 조치 방안과 보호 대책을 제시해 서버의 보안 수준을 강화할 수 있게 해준다.

1.3 연구 목적 및 주제 선정

서버에 사용되는 가장 많이 사용되는 운영체제인 Unix/Linux 계열과 MS윈도우 계열의 최신 서버 CentOS 8, Windows 10의 서버에 대하여 점검한다. 2020, 2021 KISA의 취약점 진단 가이드라인을 따라 취약점 진단을 하여 위협 요소에 대한 조치 방안과 보호 대책을 알려주어 요청자는 "Telegram Bot"을 통해 휴대폰으로 어디서든 간편하게 받아볼 수 있는 편리함을 겸비한 프로그램을 개발하려고 한다. 이를 통해 관리자는 자신의 서버의 취약한점과 보완할 점을 컴퓨터가 없어도 서버 취약점 진단을 하고 결과를 확인 할 수 있도록 하여 편리성과 효율성을 증가하도록 하는 것이 목적으로 한다. 또한 Windows 10과 CentOS 8 스크립트와 배치파일 작성을 통해 서버 운영 및 명령어를 알아보기 위해 주제로 선정하였다.

2. 관련 연구

2.1 Linux

리눅스(Linux)는 1991년 9월 17일 리누스 토르발스가 처음 출시한 운영 체제 커널인 리눅스 커널에 기반을 둔 오픈 소스 유닉스 계열 운영 체제 계열이다. 리눅스는 거의 대부분의 C언어와 어셈블리 언어로 작성되어 있어 특정 기계에 비 의존적이기 때문에 프로그램을 다른 기계의 시스템으로 포팅 하는 것이 쉬워 적합하게 변형이 가능하여 이식성과 확장성이 용이하다. 또한 텍스트 모드 중심의 관리와 다양한 관리 환경의 제공이 가능하고 모든 프로그래밍 언어를 제공하며 GNU 소프트웨어가 무료로 제공하여 소프트웨어 개발에 개방적이다. 또한 완전 무료라는 장점과, 유닉스 호환,높은 품질,기술지원, 다양한 배포 패키지를 지원하며 보안성이 높은 파일을 관리하고 시스템이 풍부한 네트워크를 지원한다.

2.2 Bash

배시(Bash)는 본 셸을 대체하는 자유 소프트웨어로서 GNU 프로젝트를 위해 Brian Fox가 작성한 유닉스 셸이다. 문법이 거의 대부분 sh와 호환되어 쓰이며 입력 중에 명령어나 파일 이름을 자동 완성해 주는 기능을 지원한다.

2.3 Python

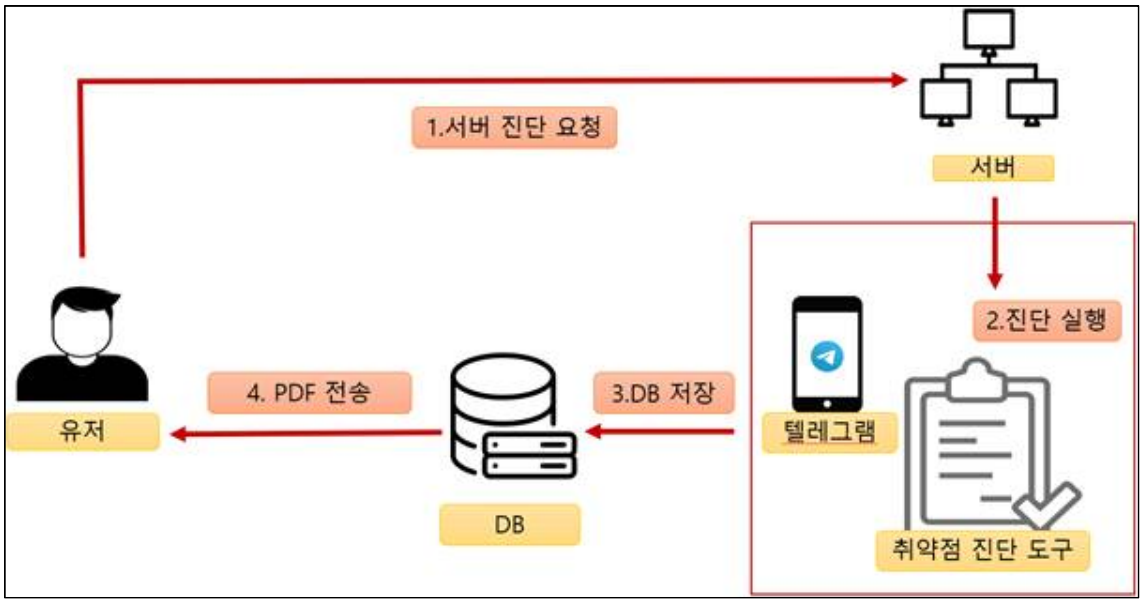
파이썬은 귀도 반 로섬이 개발한 고급 프로그래밍 언어로 플랫폼이 독립적이며 인터프리터식 객체 지향적, 동적 타이핑 대화형 언어이다. 간단하게 인터프리터 기반의 객체 지향 프로그래밍 언어라고 할 수 있다. 파이썬 프로그램은 파이썬 쉘에서 대화모드로 코드를 테스트하면서 작성할 수 있다. 빠른 개발 속도와 쉽고 간결한 문법 덕분에 파이썬은 높은 생산성을 자랑하고, 파이썬을 활용할 경우 더 적은 코드로 더 많은 작업을 수행 할 수 있으며, 복잡한 구문으로 인한 오류 발생을 줄여 그 어떤 프로그래밍 언어보다 빠른 개발이 가능하다.

2.4 Telegram-bot

Telegram-bot은 Telegram에서 제공하는 API이며 Python을 통해 서버와 연동하여 사용할 수 있으며 메시지, 명령 등을 통해 봇과 상호작용을 할 수 있다. 프로그램에 의해 운영되는 계정으로 유저의 아이디와 동등한 객체로 유저는 봇과의 대화창을 열거나 채널에 초대하여 메시지, 커맨드라인, 인라인 요청으로 봇과 상호작용하며 유저의 메시지를 읽고 쓸 수 있다.

3. 본론

3.1 시스템 구성

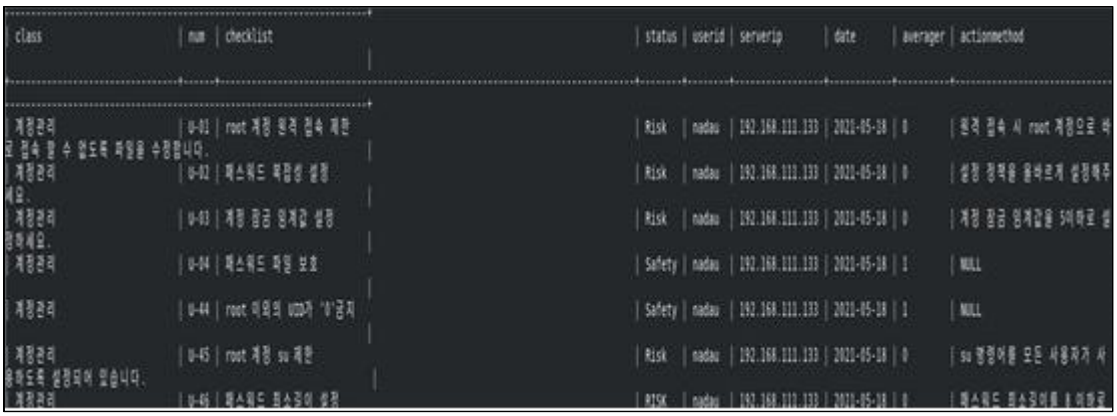


[그림 1-1] 구상도

서버 진단을 원하는 유저가 서버 진단 요청을 하면 유저의 ID, IP 등을 받아온다. 원격으로 파일을 전송해서 서버를 진단하기 위한 환경을 만들어 놓아야 하기 때문에 유저의 서버에 ssh 클라이언트가 있는지 확인해야 한다. kisa취약점 진단 가이드를 참고하여 만든 쉘 스크립트 취약점 진단도구로 진단을 실행한다. 쉘 스크립트를 ssh를 이용해서 사용자 서버에 보내고 진단을 진행한다. 마찬가지로 텔레그램 에서도 파일을 전송할 수 있게 했다. 검사가 끝난 후 결과를 텍스트 파일로 받고 DB에 연동해서 저장한다. 유저가 서버의 검사 결과, 취약, 양호 부분의 통계 값 등을 한눈에 볼 수 있도록 PDF로 저장해서 보내준다.

[그림 1-2] 진단 실행 결과

취약점 진단 후 계정 관리, 서비스 관리 등을 양호, 취약, 검사 불가인 항목이 몇 개인지 확인할 수 있게 했다. 또한 마지막 부분에는 부분을 통틀어서 검사 항목의 총 합계와 전체적으로 양호, 취약, 검사 불가 그리고 인터뷰를 진행해야 하는 항목이 몇 개인지 한 눈에 볼 수 있게 했다



[그림 1-3] 결과 저장

DB에 값을 받기 전에 먼저 DB 사용자를 먼저 만들어 주었다. 그 사용자에 관리자가 어느 서버 에서든 접속할 수 있게 권한을 주고 user와 result 테이블을 만들었다. user에는 main.sh에서 입력 받은 유저 ID와 PW 등이 먼저 들어가게 된다. user와 result에는 ID가 Primary키와 Foreign키로 연결되어 있다. main에서 ID를 입력하면 user와 result 테이블에 들어가고 그 ID를 기반으로 취약점 진단 결과가 DB에 저장된다.

```
echo -n "원격지 ID입력 :"
read RID
echo -n "원격지 IP 입력 :"
read RIP

echo $uid > userinfo
echo $RIP >> userinfo
clear

echo "폴더 생성"
ssh $RID@$RIP mkdir /jb

file1=/joongbu_script/check.sh
file2=/joongbu_script/userinfo
file3=/joongbu_script/version.txt
folder=/joongbu_script/Inspection_shell
clear
echo "스크립트 전송"
scp -r $folder $file1 $file2 $file3 $RID@$RIP:/jb

echo "스크립트 전송 완료"
rm -rf userinfo
rm -rf version.txt

sleep 1

clear
echo "스크립트 실행"
ssh -tt $RID@$RIP "sh /jb/check.sh"
```

[그림 1-4] main.sh

main.sh 코드를 실행하면 사용자 ID와 IP를 입력 받고 원격으로 폴더를 생성하여 그 폴더 안에서 취약점 진단 스크립트가 실행되게 되어 있다. 사용자 ID를 입력하면 바로 DB에 먼저 저장이 된다. check 파일도 같이 전송되게 되는데 check 파일에서 스크립트를 실행하고 결과값을 저장 받고 통계도 출력되게 했다. 결과값은 result.txt로 저장된다. 마지막에는 사용자 서버에 임의로 만들었던 파일을 지우면 검사가 끝나게 된다.

```
bot.message_handler(func=lambda message: \
                    knownUsers.get(message.chat.id).userStep == 2, commands=['sendfile'])
def send_file(message):
        try:
            client = paramiko.SSHClient()
            client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
            client.connect(hostname=knownUsers.get(message.chat.id).remoteHost, username=knownUsers.get(messag
e.chat.id).remoteUser, password=knownUsers.get(message.chat.id).remotePassword)
            scp=SCPClient(client.get_transport())
            scp.put('/joongbu_script/jb', '/', recursive=True )
            scp.close()
```

[그림 1-5] 텔레그램을 이용해 파일 전송

```
@bot.message_handler(func=lambda message: \
                     knownUsers.get(message.chat.id).userStep == 2, commands=['serverscan'])
def send_file(message):
        client = paramiko.SSHClient()
        client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
        client.connect(hostname=knownUsers.get(message.chat.id).remoteHost, username=knownUsers.get(message.c
hat.id).remoteUser, password=knownUsers.get(message.chat.id).remotePassword)
        stdin, stdout, stderr = client.exec_command("sh /jb/check.sh")
        data = stdout.read() + stderr.read()
        bot.send_message(message.chat.id, "검사완료")
        client.close()
```

[그림 1-6] 텔레그램을 이용한 서버 점검

[그림 1-7] 텔레그램 실행화면

[그림 1-8] 텔레그램 실행화면2

3.2 프로그램 구성

3.2.1 취약점 진단 프로그램

KISA의 취약점 진단 가이드를 활용하여 리눅스(CentOS) 서버 진단 쉘 스크립트와 윈도우 서버 진단 배치파일을 작성하였다. 리눅스(CentOS) 서버를 기반으로 계정 관리, 파일 및 디렉토리 관리, 서비스 관리, 패치 관리, 로그 관리 부분을 진단하였다.

리눅스(CentOS) 서버 계정 관리에서는 root 계정에 원격 접속 제한이나 패스워드의 길이, 사용 기간 등을 설정하는 진단 스크립트를 작성했다. 파일 및 디렉터리 관리에서는 파일별로 소유자나 권한 설정을 하는 진단 스크립트를 작성했다. 서비스 관리에서는 DNS나 FTP, NFS 등 여러 서비스의 활성 상태 등을 점검하는 진단 스크립트를 작성했다. 패치 관리에서는 최신 보안 패치나 권고 사항 등을 적용하는 진단 스크립트를 만들어 리눅스(CentOS) 서버의 취약점을 분석 및 평가했다.

윈도우 서버를 기반으로 계정 관리, 서비스 관리, 패치 관리, 로그 관리, 보안 관리, DB 관리 부분을 진단하였다. 윈도우 서버 계정 관리에서는 Administrator 계정 이름 변경이나 불필요한 계정 제거, 패스워드 암호 길이와 사용 기간 등을 진단하는 배치파일을 작성했다.
서비스 관리에서는 IIS 서비스와 관련한 구동 점검, CGI 실행 제한, 미사용 스크립트 매핑 제거와 FTP나 DNS 등 여러 가지 서비스 등을 점검하는 진단 배치파일을 작성했다. 패치 관리에서는 백신 프로그램 업데이트나 정책에 따른 시그템 로깅 설정 등을 하는 진단 배치파일을 작성했다. 로그 관리에서는 로그의 정기적 검토 및 보고, 이벤트 로그 관리 설정, 원격에서 이벤트 로그파일 접근 차단 등을 진단하는 배치파일을 작성했다. 보안 관리에서는 백신 프로그램 설치, 화면보호기 설정이나 시작 프로그램 목록 분석 등을 진단하는 배치파일을 작성했다. DB 관리에서는 Windows 인증 모드 사용을 진단하는 배치파일을 만들어 윈도우 서버의 취약점을 분석 및 평가했다.

3.2.1 Telegram-bot

Telegram BoT 라이브러리를 이용하여 python 언어를 사용해 Telegram과 연동하여 메시지인터페이스를 제작했다. python의 라이브러리를 이용해서 Telegram에서도 SSH접속 및 파일 전송을 가능하게 만들었다.

3.2.3 서버 및DB

윈도우와 리눅스(CentOS) 서버의 DB를 통합하여 진단 프로그램을 실행했을 때 같은 DB안에 값이 저장되게 만들었다. DB에 저장되는 항목들은 진단 관리, 진단 이름, 진단 내역, 양호/취약 상태 등이 있다.

3.2.4 PDF

윈도우와 리눅스(CentOS) 서버의 진단 프로그램을 실행했을 때 저장된 DB의 값을 각 항목별로 정리하여 PDF로 작성되게 만들었다. PDF에는 진단 관리, 진단 이름, 진단 내역, 진단 결과 값 등이 작성 되고 각 항목의 결과를 통합하여 진단하는 서버의 양호/취약의 개수를 알 수 있게 만들었다.

4. 결론

4.1 결론

리눅스(CentOS) 서버 진단 쉘 스크립트를 만들면서 이에 대한 이해도를 높이고 linux의 시스템 명령어 사용 방법과 기본적인 서버 세팅 방법 등을 익힐 수 있었고 서버의 취약한 어떠한 부분이 취약한지 알게 됐다. 윈도우 서버 진단 배치파일을 만들면서 윈도우에 대한 이해도를 높였고 윈도우의 배치파일의 명령어 사용 방법과 윈도우 서버의 특징등을 더 익힐 수 있었고 또, linux 명령어와 어떤 점이 다른지 더 자세하게 알게 됐다.

4.2 기대효과

취약점 진단 도구를 개발할 때, 모든 경우의 수를 다 따져가며 만들지 못하여 아직 정확도가 부족한 부분이 있어 추후 보완이 필요하고, 일일이 진단 요청을 확인하고 점검하고 결과를 전송하는 부분도 자동으로 진단되게끔 보완이 필요하다. 이렇게 취약점 진단 도구를 이용해 보안 상태를 점검함으로써 미리 작성해둔 배치파일을 통해 많은 양을 진단하여 시간과 인력을 최소화할 수 있고, 서버의 어느 부분이 취약한지, 보완해야 할 점이 무엇인지를 확인할 수 있고 보안 위협에 대응하는 완벽한 체계를 구축할 수 있다. 또한, 스마트폰으로도 빠른 진단이 가능하여 관리자가 자리에 없어도 진단 요청이 들어오면 관리자는 빠르게 취약점 진단을 할 수 있어 효율성과 편리성이 증가한다.

5. 별첨

5.1 소스코드

main.sh

```
#!/bin/bash

clear

while    [ True ]
do
        echo -e "₩n********************취약점    진단 프로그램********************₩n"
        echo -e "                                                  만든이 : 2조(이게뭐조)₩n"
        echo          -en          "1.진단하기₩n₩n2.진단결과          확인하기₩n₩n3.만든이
보기₩n₩n4.나가기₩n₩n입력:"
        read check


        if [ $check -eq 1 ]; then


        while [ True ]
        do

        echo    -n "회원ID 입력 :"
        read    TMPID

        mysql      -h localhost -unawon -p1234 -e "select id from user where id like
'$TMPID';" joongbu_db > /joongbu_script/UID.txt

        if [    -z "`cat /joongbu_script/UID.txt`" ]; then
                clear
        echo    "존재하지 않는 아이디 입니다."
        continue
        else
        clear
        echo    "아이디가 존재 합니다."
        uid=`cat    /joongbu_script/UID.txt | tail -1`
        #rm    -rf /joongbu_script/UID.txt
        break
        fi
        done


        function linux() {
        clear
        python3    /joongbu_script/server_version.py

        echo    -n "원격지 ID입력 :"
        read    RID
        echo    -n "원격지 IP 입력 :"
        read    RIP
```

```
echo    $uid >> /joongbu_script/userinfo
echo    $RIP >> /joongbu_script/userinfo
clear

echo    "폴더 생성"
ssh    $RID@$RIP mkdir /jb

file1=/joongbu_script/check.sh
file2=/joongbu_script/userinfo
file3=/joongbu_script/version.txt
file4=/joongbu_script/check1.sh
folder=/joongbu_script/Inspection_shell
clear
echo    "스크립트 전송"
scp -r    $folder $file1 $file2 $file3 $file4 $RID@$RIP:/jb

echo    "스크립트 전송 완료"
#rm    -rf userinfo
#rm    -rf version.txt

sleep    1

clear
echo    "스크립트 실행"
ssh    -tt $RID@$RIP "sh /jb/check.sh"
}


function windows()    {
clear

echo    -n "원격지 ID입력 :"
read    RID
        echo -n "원격지 IP 입력 :"
         read    RIP

         echo    "폴더 생성"
         ssh    $RID@$RIP mkdir C:₩jb

file1=/joongbu_script/w_check.bat
          folder=/joongbu_script/Windows_Inspection_shell
         clear
         echo    "스크립트 전송"
         scp    -r $folder $file1 $RID@$RIP:C:₩jb

         echo    "스크립트 전송 완료"

         sleep    1

         clear
         echo    "스크립트 실행"
         ssh    $RID@$RIP "C:₩Users₩$RID₩jb₩w_check.bat"
```

```
        }


        while [ True ]
        do
                echo     -en "₩nOS 종류를 선택 하세요. (1. linux(centos) 2.     windows
server 2012) : "
                read    OS_version


                if    [ $OS_version -eq 1 ]; then
              linux
        break
        elif [    $OS_version -eq 2 ]; then
        windows
        break
        else
        echo    "₩e[1;31m값이 잘못되었습니다. 다시 입력해주세요.₩e[0m"
        continue
                fi
        done
        sleep 2
        clear
        continue


        elif [ $check -eq 2 ]; then
        sh /jb/check1.sh
        echo "취약점    진단 프로그램으로 돌아갑니다."
        read -p "메뉴로    돌아가시겠습니까? (y):" RESP
                if [ "$RESP" =    "y" ]; then
                        echo "메뉴로    돌아갑니다."
                        clear
                        continue
        fi

        elif [ $check -eq 3 ]; then
        clear
        echo "   < 중부대학교 고양캠퍼스 소속    >"
                echo -e "₩n91613703 남승택                                91812335 박세빈₩n
₩n91812608 신자연                          91812701 왕나원₩n ₩n91812672 양재희₩n"

                echo -e "₩n   < 주요 프로젝트 내용    >"
                echo -e "₩n리눅스 서버    구축 및 취약점 진단 스크립트 작성"
                echo -e "₩n윈도우 서버    구축 및 취약점 진단 스크립트 작성"
                echo -e "₩n진단 결과 pdf 파일로 확인"
                echo -e "₩nDB 연동 및    텔레그램 연동₩n"

        read -p "메뉴로    돌아가시겠습니까? (y):" RESP
        if [    "$RESP" = "y" ]; then
                        echo "메뉴로    돌아갑니다."
                        clear
```

```
        continue
        fi


        elif [ $check -eq 4 ]; then
        read -p "취약점    진단 프로그램을 나가시겠습니까? (y/n):" RESP
        if [    "$RESP" = "y" ]; then
        echo    "취약점 진단 프로그램을 나갑니다."
        break
        elif [    "$RESP" = "n" ]; then
        echo    "취약점 진단 프로그램을 계속합니다."
        sleep    3
        clear
        continue
        else
        echo    -e "₩e[1;31m값이 잘못되었습니다. 다시 입력해주세요.₩e[0m"
        sleep    3
        clear
        continue
        fi
        fi
done




#rm    -rf /jb
```

check.sh

리눅스(CentOS) 서버 진단 스크립트 : U-07

```
#!/bin/bash

echo    ""
echo    "< U-07 /etc/passwd 파일 소유자 및 권한 설정 >    "
echo    ""

file="/etc/passwd"


if [ -f    "/etc/passwd" ]; then

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

        if [ "$owner" ==    "root" ]; then
```

```
        if [    "$perm" == "644" ] || [ "$perm" ==    "640" ] || [ "$perm" == "600" ] || [
"$perm" == "444" ] || [ "$perm" ==    "440" ] || [ "$perm" == "400" ]; then
                        echo -e "상태 : ₩e[1;36m양호₩e[0m"
                        echo "파일 및 디렉터리 관리,U-07,/etc/passwd 파일    소유자 및
권한 설정,Safety,$userid,$serverip,$DATE,1" >>    /jb/result.txt
        FileDirSafety=`expr    $FileDirSafety + 1`

                elif    [ "$perm" != "644" ] || [ "$perm" !=    "640" ] || [ "$perm" != "600"
] || [    "$perm" != "444" ] || [ "$perm" !=    "440" ] || [ "$perm" != "400" ]; then
                        echo -e "상태 : ₩e[1;31m취약₩e[0m"
                        echo    "/etc/passwd 파일의 권한이 644이하가 아닙니다."
                        echo "파일 및    디렉터리 관리,U-07,/etc/passwd 파일 소유자 및
권한 설정,Risk,$userid,$serverip,$DATE,0,/etc/passwd 파일의 권한이 644이하가 아닙니다." >>
/jb/result.txt
        FileDirRisk=`expr    $FileDirRisk + 1`

        fi
        elif [ "$owner" !=    "root" ] && [ "$perm" == "644" ] || [    "$perm" == "640" ] ||
[ "$perm" ==    "600" ] || [ "$perm" == "444" ] || [    "$perm" == "440" ] || [ "$perm" ==
 "400" ]; then
        #if [    "$perm" == "644" ] || [ "$perm" ==    "640" ] || [ "$perm" == "600" ] || [
 "$perm" == "444" ] || [ "$perm" ==    "440" ] || [ "$perm" == "400" ]; then
        echo    -e "상태 : ₩e[1;31m취약₩e[0m"
                        echo    "/etc/passwd 파일의 소유자가 root가 아닙니다."
                        echo "파일 및    디렉터리 관리,U-07,/etc/passwd 파일 소유자 및
권한 설정,Risk,$userid,$serverip,$DATE,0,/etc/passwd 파일의 소유자가 root가 아닙니다." >>
/jb/result.txt
        FileDirRisk=`expr    $FileDirRisk + 1`

        else
        echo -e "상태 : ₩e[1;31m취약₩e[0m"
                echo "/etc/passwd 파일의    소유자가 root가 아니고 권한이 644이하가 아닌
경우입니다."
                echo "파일 및 디렉터리 관리,U-07,/etc/passwd 파일 소유자 및 권한
설정,Risk,$userid,$serverip,$DATE,0,/etc/passwd    파일의 소유자가 root가 아니고 권한이
644이하가    아닙니다." >> /jb/result.txt
        FileDirRisk=`expr    $FileDirRisk + 1`

        fi

else
        echo -e "상태 : ₩e[1;33m검사불가₩e[0m"
        echo "/etc/passwd 가 존재 하지    않습니다."
        echo    "파일    및    디렉터리    관리,U-07,/etc/passwd    파일    소유자    및    권한
설정,Error,$userid,$serverip,$DATE,0,/etc/passwd가    존재 하지 않습니다." >> /jb/result.txt
        FileDirError=`expr $FileDirError    + 1`

fi

echo    ""
e                               c                               h                               o
================================================================================
==================
```

리눅스(CentOS) 서버 진단 스크립트 : U-08

```
#!/bin/bash

echo    ""
echo    "< U-08 /etc/shadow 파일 소유자 및 권한 설정 >    "
echo    ""

file="/etc/shadow"

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

root=`    echo "$perm" | awk '{print substr($0,1,1)}' `
group=`    echo "$perm" | awk '{print substr($0,2,1)}' `
other=`    echo "$perm" | awk '{print substr($0,3,1)}' `

if [ -f    "/etc/shadow" ] ;
then
        if [ "$owner" ==    "root" ] && [[ "$root" -eq 4 ]] && [[    "$group" -eq 0 ]] &&
[[ "$other" -eq 0 ]]; then
                    echo -e "상태 : ₩e[1;36m양호₩e[0m"
                    echo "파일 및 디렉터리 관리,U-08,/etc/shadow 파일 소유자 및 권한
설정,Safety,$userid,$serverip,$DATE,1,/etc/shadow     파일의 소유자가 root이고 권한이 400인
경우입니다." >> /jb/result.txt
        FileDirSafety=`expr    $FileDirSafety + 1`


        elif [ "$owner" !=    "root" ] && [[ "$root" -eq 4 ]] && [[    "$group" -eq 0 ]] &&
[[ "$other" -eq 0 ]]; then
                            echo -e "상태 : ₩e[1;31m취약₩e[0m"
                            echo    "/etc/shadow 파일의 소유자가 root가 아닌 경우입니다."
                            echo "파일 및     디렉터리 관리,U-08,/etc/shadow 파일 소유자 및
권한     설정,Risk,$userid,$serverip,$DATE,0,/etc/passwd     파일의     소유자가     root가     아닌
경우입니다." >>     /jb/result.txt
        FileDirRisk=`expr    $FileDirRisk + 1`


        elif [ "$owner" ==    "root" ] && [[ "$root" != 4 ]] && [[    "$group" != 0 ]] &&
[[ "$other" != 0 ]]; then
                            echo -e "상태 : ₩e[1;31m취약₩e[0m"
                            echo "/etc/shadow 파일의 권한이 400이 아닌 경우입니다."
                            echo "파일 및     디렉터리 관리,U-08,/etc/shadow 파일 소유자 및
권한 설정,Risk,$userid,$serverip,$DATE,0,/etc/shadow 파일의 권한이 400이 아닌 경우입니다."
>>     /jb/result.txt
        FileDirRisk=`expr    $FileDirRisk + 1`


        else
                            echo -e "상태 : ₩e[1;31m취약₩e[0m"
                            echo    "/etc/shadow 파일의 소유자가 root가 아니고 권한이 400이
아닌 경우입니다."
```

```
                        echo "파일 및  디렉터리 관리,U-08,/etc/shadow 파일 소유자 및 권한 설정,Risk,$userid,$serverip,$DATE,0,/etc/shadow 파일의 소유자가 root가 아니고 권한이 400이 아닌 경우입니다." >> /jb/result.txt
        FileDirRisk=`expr  $FileDirRisk + 1`

        fi

else
        echo -e "상태 : ₩e[1;33m검사불가₩e[0m"
        echo "/etc/shadow 가 존재 하지  않습니다."
        echo  "파일 및 디렉터리 관리,U-08,/etc/shadow 파일 소유자 및 권한 설정,Error,$userid,$serverip,$DATE,0,/etc/shadow  가 없습니다." >> /jb/result.txt
        FileDirError=`expr $FileDirError  + 1`

fi

echo  ""
e                       c                       h                       o
================================================================================
```

리눅스(CentOS) 서버 진단 스크립트 : U-09

```
#!/bin/bash

echo  ""
echo  "< U-09 /etc/hosts 파일 소유자 및 권한 설정 > "
echo  ""

if [ -f  "/etc/hosts" ] ; then

        file="/etc/hosts"
        Autority=$(ls -l $file | awk  '{print $1}' | cut -c 2-10)
        owner=` stat -c %U $file `
        perm=` stat -c %a $file `

        root=` echo "$perm" |  awk '{print substr($0,1,1)}' `
        group=` echo "$perm" |  awk '{print substr($0,2,1)}' `
        other=` echo "$perm" |  awk '{print substr($0,3,1)}' `


        if [ "$owner" ==  "root" ] && [ "$root" -eq 6 ] && [  "$group" -eq 0 ] && [ "$other" -eq 0 ]; then
        echo -e "상태 : ₩e[1;36m양호₩e[0m"
                echo  "파일 및 디렉터리 관리,U-09,/etc/hosts 파일 소유자 및 권한 설정,Safety,$userid,$serverip,$DATE,1,/etc/hosts파일의  소유자가 root이고 권한이 600인 경우입니다." >> /jb/result.txt
        FileDirSafety=`expr  $FileDirSafety + 1`
```

```
        elif  [ "$owner" != "root" ]   && [ "$root" -eq 6 ] && [ "$group" -eq 0   ] && [ "$other" -eq 0 ]; then
        echo -e "상태 : ₩e[1;31m취약₩e[0m"
                echo "/etc/hosts파일의   소유자가 root아닌 경우입니다."
                echo "파일 및 디렉터리 관리,U-09,/etc/hosts 파일 소유자 및 권한 설정,Risk,$userid,$serverip,$DATE,0,/etc/hosts파일의   소유자가 root가 아닌 경우입니다." >> /jb/result.txt
        FileDirRisk=`expr   $FileDirRisk + 1`



        elif [ "$owner" ==   "root" ] && [ "$Autority" !=   "rw-------" ]; then
        echo -e "상태 : ₩e[1;31m취약₩e[0m"
                echo "/etc/hosts파일의   권한이 600이 아닌 경우입니다."
                echo "파일 및 디렉터리 관리,U-09,/etc/hosts 파일 소유자 및 권한 설정,Risk,$userid,$serverip,$DATE,0,/etc/hosts파일의   권한이 600이 아닌 경우입니다." >> /jb/result.txt
        FileDirRisk=`expr   $FileDirRisk + 1`



        else
                echo -e "상태 : ₩e[1;31m취약₩e[0m"
                echo "/etc/hosts파일의   소유자가 root가 아니고   권한이 600이 아닌 경우입니다."
                echo "파일 및 디렉터리 관리,U-09,/etc/hosts 파일 소유자 및 권한 설정,Risk,7$userid,$serverip,$DATE,0,/etc/hosts파일의   소유자가 root가 아니고 권한이 600이 아닌 경우입니다." >> /jb/result.txt
        FileDirRisk=`expr   $FileDirRisk + 1`

    fi


else
        echo -e "상태 : ₩e[1;33m검사불가₩e[0m"
        echo "/etc/hosts가 존재 하지   않습니다."
        echo "파일 및 디렉터리 관리,U-09,/etc/hosts 파일 소유자 및 권한 설정,Error,$userid,$serverip,$DATE,0,/etc/hosts가   없습니다." >> jb/result.txt
        FileDirError=`expr $FileDirError   + 1`

fi

echo   ""
e                               c                               h                               o
================================================================================
```

```
#!/bin/bash

echo    ""
echo    "< U-10 /etc/(x)inetd.conf 파일 소유자 및 권한 설정    > "
echo    ""

ls    /etc/xinetd.d  >  /jb/file.txt

if [ ! -f    "/etc/xinetd,conf" ] && [ ! -f "/etc/inetd.conf"    ] && [ ! -s "/jb/file.txt" ]; then
        echo -e "상태 : ₩e[1;33m검사불가₩e[0m"
        echo    "/etc/(x)inetd.conf,/etc/inetd.conf,/etc/inetd.d가 존재하지 않습니다."
        echo "파일 및 디렉터리 관리,U-10,/etc/(x)inetd.conf 파일 소유자 및 권한 설정,Erorr,$userid,$serverip,$DATE,0,/etc/(x)inetd.conf,/etc/inetd.conf,/etc/inetd.d가 존재하지 않습니다." >> /jb/result.txt
        FileDirError=`expr $FileDirError    + 1`

        echo ""
        e                               c                               h                               o
==========================================================================
==================
        return
        #exit

fi


flag=0

if [ -f    "/etc/xinetd.conf" ] ; then
        file="/etc/xinetd.conf"
        perm=` stat -c %a $file `
        owner=` stat -c %U $file `

        root=` echo "$perm" |    awk '{print substr($0,1,1)}' `
        group=` echo "$perm" |    awk '{print substr($0,2,1)}' `
        other=` echo "$perm" |    awk '{print substr($0,3,1)}' `

        Autority=$(ls -l $file | awk    '{print $1}' | cut -c 2-10)

        if [ "$owner" ==    "root" ] && [ "$root" -eq 6 ] && [    "$group" -eq 0 ] && [ "$other" -eq 0 ]; then
        echo ""    > /dev/null

        elif  [ "$owner" != "root" ]    && [ "$root" -eq 6 ] && [ "$group" -eq 0    ] && [ "$other" -eq 0 ]; then
        flag=` expr $flag +    1 `
                echo  "/etc/xinetd,conf  파일의       소유자가  root아닌  경우입니다." >> /jb/tmp.txt

        elif [ "$owner" ==    "root" ] && [ "$Autority" !=    "rw-------" ]; then
        flag= ` expr $flag    + 1 `
```

```
                echo "/etc/xinetd.conf파일의 권한이 600이 아닌 경우입니다." >>
/jb/tmp.txt

        else
        flag= `expr $flag +  1 `
                echo "/etc/xinetd.conf 파일의 소유자가 root가 아니거나 권한이 600인
경우가 아닙니다." >> /jb/tmp.txt

        fi

else
        flag=` expr $flag + 1 `
        echo "/etc/xinetd.conf가  존재 하지 않습니다." >> /jb/error.txt
fi

if [ -f  "/etc/inetd.conf" ] ; then
        file1="/etc/inetd.conf"
        perm=` stat -c %a $file1 `
        owner=` stat -c %U $file1 `

        root=` echo "$perm" | awk  '{print substr($0,1,1)}' `
        group=` echo "$perm" | awk  '{print substr($0,2,1)}' `
        other=` echo "$perm" | awk  '{print substr($0,3,1)}' `

        Autority=$(ls -l $file1 | awk '{print  $1}' | cut -c 2-10)

        if [ "$owner" ==  "root" ] && [ "$root" -eq 6 ] && [  "$group" -eq 0 ] && [
"$other" -eq 0 ]; then
        echo ""  > /dev/null

        elif  [ "$owner" != "root" ] && [  "$root" -eq 6 ] && [ "$group" -eq 0 ] &&
 [ "$other" -eq 0 ]; then
        flag=` expr $flag +  1 `
                echo "/etc/inetd,conf 파일의 소유자가 root아닌 경우입니다." >>
/jb/tmp.txt

        elif [ "$owner" ==  "root" ] && [ "$Autority" !=  "rw-------" ]; then
                flag=` expr $flag + 1 `
        echo  "/etc/inetd.conf파일의 권한이 600이 아닌 경우입니다." >> /jb/tmp.txt

        else
        flag= ` expr $flag  + 1 `
                echo "/etc/inetd.conf 파일의 소유자가 root가 아니거나 권한이 600인
경우가 아닙니다." >> /jb/tmp.txt

        fi

else
        flag=` expr $flag + 1 `
         echo "/etc/inetd.conf가 존재  하지 않습니다." >> /jb/error.txt
fi


cnt=0
```

```
if [ -d    "/etc/xinetd.d" ] ; then
          file2="/etc/xinetd.d"
          for i in `ls $file2`
          do

          perm=` stat -c %a    /etc/xinetd.d/$i `
          owner=` stat -c %U    /etc/xinetd.d/$i `
          if [    "$owner" == "root" ] && [ "$perm" ==    600 ]; then
          echo    "" > /dev/null
          elif [    "$owner" != "root" ] && [ "$perm" ==    600 ]; then
          cnt=`expr    $cnt + 1`
          echo 파일명: $i / 소유자: $owner / 권한: $perm >> /jb/tmp.txt
          elif [    "$owner" == "root" ] && [ "$perm" !=    600 ]; then
          cnt=`expr    $cnt + 1`
          echo 파일명: $i / 소유자: $owner / 권한: $perm >> /jb/tmp.txt
          else
          cnt=`expr    $cnt + 1`
          echo 파일명: $i / 소유자: $owner / 권한: $perm >> /jb/tmp.txt
          fi
          done

          if [ $cnt = 0 ]; then
          echo ""    > /dev/null
          else
                    echo "/etc/xinetd.d 파일의       소유자가 root가 아니거나 권한이 600인
경우가 아닙니다." >> /jb/tmp.txt
#         cat /jb/tmp.txt
          fi
else
          flag=` expr $flag + 1 `
          echo "/etc/xinetd.d가 존재 하지    않습니다." >> /jb/error.txt

fi

unset cnt

if [    "$flag" -eq 0 ]; then
          echo -e "상태 : ₩e[1;36m양호₩e[0m"
          echo    "파일    및    디렉터리    관리,U-10,/etc/(x)inetd.conf    파일    소유자    및    권한
설정,Safety,$userid,$serverip,$DATE,1"    >> /jb/result.txt
          FileDirSafety=`expr    $FileDirSafety + 1`

else
          echo -e "상태 : ₩e[1;31m취약₩e[0m"
          if [ -f /jb/tmp.txt ]; then
          cat /jb/tmp.txt
          fi
          if [ -f /jb/error.txt ]; then
          cat /jb/error.txt
          fi
```

```
        echo "파일 및 디렉터리 관리,U-10,/etc/(x)inetd.conf 파일 소유자 및 권한 설정,Safety,$userid,$serverip,$DATE,0,/etc/(x)inetd.conf 파일 소유자 및 권한 설정" >> /jb/result.txt
        FileDirRisk=`expr $FileDirRisk + 1`

fi


rm -rf /jb/tmp.txt /jb/error.txt


echo ""
e                    c                    h                    o
================================================================================
```

윈도우(Windows 10) 서버 진단 스크립트 : W-check

```
@ECHO OFF


if not "%1"=="am_admin" (powershell start -verb runas '%0' am_admin & exit /b)

ECHO.
ECHO ************************************** 취약점 진단 프로그램 **************************************

ECHO.

ECHO 만든이 : 2조(이게뭐조)

ECHO 1. 진단 하기

ECHO.

::ECHO 2. 진단 결과 확인하기

ECHO.

ECHO 2. 나가기

ECHO.

::ECHO 4. PDF

E                    C                    H                    O
********************************************************************************************************
ECHO.

ECHO.
```

```
set    /p x=입력 :

if    %x%==1 goto first

if    %x%==2 goto second

if    %x%==3 goto third

::set    /p userid=userid insert :
::set    /p serverip=serverip insert :

::set    dir=C:₩jb₩Windows_Inspection_shell

::set    /p ID=DB ID insert :
::set    /p PASSWD=DB passwd insert

set    AccountSafety=0
set    AccountRisk=0
set    AccountError=0

set    ServiceSafety=0
set    ServiceRisk=0
set    ServiceError=0

set SecuritySafety=0
set    SecurityRisk=0
set    SecurityError=0


set    LogSafety=0
set    LogRisk=0
set    LogError=0


set    PatchSafety=0
set    PatchRisk=0
set    PatchError=0

cls

:first

cls

set    /p userid=userid insert :
set    /p serverip=serverip insert :


set    dir=C:₩jb₩Windows_Inspection_shell

set    /p ID=DB ID insert :
set    /p PASSWD=DB passwd insert

echo    %serverip%의 서버를 점검 합니다.
```

```
pause
cls

echo                              계정관리 검사
call    %dir%₩W-01.bat
call    %dir%₩W-02.bat
call    %dir%₩W-03.bat
call    %dir%₩W-04.bat
call    %dir%₩W-05.bat
call    %dir%₩W-06.bat
call    %dir%₩W-46.bat
call    %dir%₩W-47.bat
call    %dir%₩W-48.bat
call    %dir%₩W-49.bat
call    %dir%₩W-50.bat
call    %dir%₩W-51.bat
call    %dir%₩W-52.bat
call    %dir%₩W-53.bat
call    %dir%₩W-54.bat
call    %dir%₩W-55.bat
call    %dir%₩W-56.bat
call    %dir%₩W-57.bat

echo 계정관리    양호 : %AccountSafety%
echo 계정관리    취약 : %AccountRisk%
echo 계정관리    검사 불가 : %AccountError%
echo 계정관리    자체검사 : 1
echo.    & echo. & echo. & echo. & echo.



echo                              서비스 관리 검사
call    %dir%₩W-07.bat
call    %dir%₩W-08.bat
call    %dir%₩W-09.bat
call    %dir%₩W-10.bat
call    %dir%₩W-11.bat
call    %dir%₩W-12.bat
call    %dir%₩W-13.bat
call    %dir%₩W-14.bat
call    %dir%₩W-15.bat
call    %dir%₩W-16.bat
call    %dir%₩W-17.bat
call    %dir%₩W-18.bat
call    %dir%₩W-19.bat
call    %dir%₩W-20.bat
call    %dir%₩W-21.bat
call    %dir%₩W-22.bat
call    %dir%₩W-23.bat
call    %dir%₩W-24.bat
call    %dir%₩W-25.bat
call    %dir%₩W-26.bat
call    %dir%₩W-27.bat
call    %dir%₩W-28.bat
```

```
call    %dir%₩W-29.bat
call    %dir%₩W-30.bat
call    %dir%₩W-31.bat
call    %dir%₩W-58.bat
call    %dir%₩W-59.bat
call    %dir%₩W-60.bat
call    %dir%₩W-61.bat
call    %dir%₩W-62.bat
call    %dir%₩W-63.bat
call    %dir%₩W-64.bat
call    %dir%₩W-65.bat
call    %dir%₩W-66.bat
call    %dir%₩W-67.bat
call  %dir%₩W-68.bat

echo 서비스     관리 양호 : %ServiceSafety%
echo 서비스     관리 취약 : %ServiceRisk%
echo 서비스     관리 검사불가 : %ServiceError%
echo 서비스     관리 자체검사 : 1
echo.    & echo. & echo. & echo. & echo.


echo                            패치관리 검사
call    %dir%₩W-32.bat
call    %dir%₩W-33.bat
call    %dir%₩W-69.bat

echo 패치관리    양호 : %PatchSafety%
echo 패치관리    취약 : %PatchRisk%
echo 패치관리    검사불가 : %PatchError%
echo 패치관리    자체검사 : 1


echo                            로그관리 검사
call    %dir%₩W-34.bat
call    %dir%₩W-35.bat
call    %dir%₩W-70.bat

echo 로그관리    양호 : %LogSafety%
echo 로그관리    취약 : %LogRisk%
echo 로그관리    검사불가 : %LogError%
echo 로그관리    자체검사 : 1




echo                            보안관리 검사
call    %dir%₩W-36.bat
call    %dir%₩W-37.bat
call    %dir%₩W-38.bat
call    %dir%₩W-39.bat
call    %dir%₩W-40.bat
call    %dir%₩W-41.bat
call    %dir%₩W-42.bat
```

```
call    %dir%₩W-43.bat
call    %dir%₩W-44.bat
call    %dir%₩W-45.bat
call    %dir%₩W-72.bat
call    %dir%₩W-73.bat
call    %dir%₩W-74.bat
call    %dir%₩W-75.bat
call    %dir%₩W-76.bat
call    %dir%₩W-77.bat
call    %dir%₩W-78.bat
call    %dir%₩W-79.bat
call    %dir%₩W-80.bat
call    %dir%₩W-81.bat

echo 보안관리    양호 : %SecuritySafety%
echo 보안관리    취약 : %SecurityRisk%
echo 보안관리    검사불가 : %SecurityError%
echo 보안관리    자체검사 : 1

echo    ---------- 진단을 완료 했습니다    ----------

goto start

echo.    & echo. & echo. & echo. & echo.
pause
mysql     -h 192.168.111.100 -unawon -p1234 -e "load data local infile     'C:/jb/result.txt' into table result fields terminated by ',';"    joongbu_db
rmdir    /S /Q C:₩Users₩Administrator₩jb₩
exit
```

윈도우(Windows 10) 서버 진단 스크립트 : W-01

```
@ECHO OFF


ECHO.


ECHO ^< W-01 Administrator 계정 이름 변경 또는 보안성 강화 ^>


ECHO.




net user | find /i "Administrator" > C:₩w01.txt


IF %errorlevel% EQU 0 (echo "상태 : 취약"&echo."Administrator의 계정 이름이 설정되어 있지 않습니다."
```

```
                    echo "계정 관리,W-01,Administrator 계정 이름 변경 또는 보안성
강화,Risk,%userid%,%serverip%,%date%,0,Administrator의    계정 이름을 설정하세요." >>
C:₩Users₩%name%₩jb₩result.txt


                    set /a AccountRisk=%AccountRisk%+1


) ELSE (echo "상태 : 양호"


echo "계정 관리,W-01,Administrator 계정    이름 변경 또는 보안성
강화,Safety,%userid%,%serverip%,%date%,1"    >> C:₩Users₩%name%₩jb₩result.txt


set /a AccountSafety=%AccountSafety%+1


)




del C:₩w01.txt


ECHO.

ECHO.=============================================================
======================
```

윈도우(Windows 10) 서버 진단 스크립트 : W-02

```
@ECHO OFF


ECHO.


ECHO ^< W-02 Guest 계정 비활성화 ^>


ECHO.




net user guest | find "활성 계정" |    find "예" > C:₩w02.txt
```

```
IF %errorlevel% EQU 0 (echo "상태 : 취약"&echo."Guest 계정이 활성화 되어있습니다."

                    echo "계정 관리,W-02,Guest 계정
비활성화,Risk,%userid%,%serverip%,%date%,0,Guest     계정을 비활성화하세요." >>
C:₩Users₩%name%₩jb₩result.txt

                    set /a AccountRisk=%AccountRisk%+1

) ELSE (echo "상태 : 양호"

echo "계정 관리,W-02,Guest 계정 비활성화,Safety,%userid%,%serverip%,%date%,1" >>
C:₩Users₩%name%₩jb₩result.txt

set /a AccountSafety=%AccountSafety%+1

)

del C:₩w02.txt

ECHO.

ECHO.=============================================================
======================
```

윈도우(Windows 10) 서버 진단 스크립트 : W-03

```
@ECHO OFF

ECHO.

ECHO ^< W-03 불필요한 계정 제거 ^>

ECHO.

echo *계정현황 : 불필요한 계정을 확인하세요*
```

```
net user




echo "계정 관리,W-03,불필요한 계정 제거,N/A,%userid%,%serverip%,%date%,1" >>
C:₩Users₩%name%₩jb₩result.txt


set /a AccountSafety=%AccountSafety%+1




ECHO.

ECHO.=============================================================
======================
```

윈도우(Windows 10) 서버 진단 스크립트 : W-04

```
@ECHO OFF


ECHO.


ECHO ^< W-04 계정 잠금 임계값 설정 ^>


ECHO.




net accounts | find "잠금 임계값"    > C:₩w04.txt


FOR /f "tokens=1-3" %%a IN (w04.txt) DO SET value=%%c


IF %value% GEQ 6 (echo "상태 : 취약"&echo."계정 임계값이 6이상으로    설정되어
있습니다."
```

```
                echo "계정 관리,W-04,계정 잠금 임계값
설정,Risk,%userid%,%serverip%,%date%,0,계정    잠금 임계값을 5이하로 설정하세요." >>
C:₩Users₩%name%₩jb₩result.txt


                set /a AccountRisk=%AccountRisk%+1


) ELSE IF %value% EQU 0 (echo "상태 : 취약"&echo."계정 임계값이 설정되어 있지
않습니다."


                echo "계정 관리,W-04,계정 잠금 임계값
설정,Risk,%userid%,%serverip%,%date%,0,계정    잠금 임계값을 5이하로 설정하세요." >>
C:₩Users₩%name%₩jb₩result.txt


                set /a AccountRisk=%AccountRisk%+1


) ELSE IF %value%=="아님" (echo    "상태 : 취약"&echo."계정    임계값이 설정되어 있지
않습니다."


                echo "계정 관리,W-04,계정 잠금 임계값
설정,Risk,%userid%,%serverip%,%date%,0,계정    잠금 임계값을 5이하로 설정하세요." >>
C:₩Users₩%name%₩jb₩result.txt


                set /a AccountRisk=%AccountRisk%+1


) ELSE (echo "상태 : 양호"


echo "계정 관리,W-04,계정 잠금 임계값 설정,Safety,%userid%,%serverip%,%date%,1" >>
C:₩Users₩%name%₩jb₩result.txt


set /a AccountSafety=%AccountSafety%+1


)




del C:₩w04.txt


ECHO.
```

```
ECHO.==========================================================
======================
```

윈도우(Windows 10) 서버 진단 스크립트 : W-05

```
@ECHO OFF

ECHO.

ECHO ^< W-5 해독 가능한 암호화를 사용하여 암호 저장 기준    ^>

ECHO.



secedit /export /cfg C:₩jb₩LocalSecurityPolicy.txt | find /v    "작업을" | find /v "자세한"



TYPE C:₩jb₩LocalSecurityPolicy.txt | find /i    "ClearTextPassword" | find "0" > NUL



if not errorlevel 1 echo "결과 : 양호"

if not errorlevel 1 echo "계정 관리,W-05,해독    가능한 암호화를 사용하여 암호
저장기준,Safety,%userid%,%serverip%,%date%,1"    >> C:₩Users₩%name%₩jb₩result.txt

if not errorlevel 1 set /a AccountSafety=%AccountSafety%+1



if errorlevel 1 echo "결과 : 취약"&echo."해독 가능한 암호화를 사용하여 암호 저장 정책이
사용으로 되어 있습니다."
```

```
if errorlevel 1 echo "계정 관리,W-05,해독    가능한 암호화를 사용하여 암호
저장기준,Risk,%userid%,%serverip%,%date%,0,해독    가능한 암호화를 사용하여 암호 저장
정책을 사용 안 함으로 설정하세요." >>    C:₩Users₩%name%₩jb₩result.txt


if errorlevel 1 set /a AccountRisk=%AccountRisk%+1




ECHO.

ECHO.============================================================
======================
```

텔레그램을 통한 점검 : bot.py

```
# -*-    coding: utf-8 -*-
import    config
import    allMessage
import    telebot
import    paramiko
import    os
import    sys
import    time
import    pymysql
import    re
from    scp  import  SCPClient

bot  =    telebot.TeleBot(config.botToken)

class    User:
     def  __init__(self,  userChatId):
          self.userChatId  =  userChatId

     def  userName(self,  userName):
          self.userName  =  userName

     def  sshUser(self,  sshUser):
          self.sshUser  =  sshUser

     def  sshPassword(self,  sshPassword):
          self.sshPassword  =  sshPassword

     def  sshHost(self,  sshHost):
          self.sshHost  =  sshHost

     def  remoteUser(self,remoteUser):
          self.remoteUser  =  remoteUser
```

```
    def remotePassword(self, remotePassword):
        self.remotePassword = remotePassword

    def remoteHost(self, remoteHost):
        self.remoteHost = remoteHost

    def clientid(self, clientid):
        self.clientid = clientid

    def cdCommand(self, cdCommand):
        self.cdCommand = cdCommand

    def userStep(self, userStep):
        self.userStep = userStep
        #(If user for chatId) = None - user    not activate
        #step = 1 - wait everything
        #step = 2 - activ

knownUsers    = {}


#
#
#Pass    all message(exclude /start, /on, /help), if user not activate:
@bot.message_handler(func=lambda    message: ₩
                         ((knownUsers.get(message.chat.id)          ==          None)          or
(knownUsers.get(message.chat.id) == 1)) ₩
                        and (message.text !=    '/start') and (message.text != '/on') ₩
                        and (message.text !=    '/help'), content_types=["text"])
def    pass_message(message):
    pass

#
#
#Message    /start after and before register user
@bot.message_handler(func=lambda    message: ₩
                         knownUsers.get(message.chat.id) == None, commands=['start'])
def    send_welcome(message):
    bot.send_message(message.chat.id,    allMessage.commands['start_AfterAuthorized'])

@bot.message_handler(func=lambda    message: ₩
                         knownUsers.get(message.chat.id).userStep == 2, commands=['start'])
def    send_welcome(message):
    bot.send_message(message.chat.id,    allMessage.commands['start_BeforeAuthorized'])
#
#
#Message    /help after and before register user
@bot.message_handler(func=lambda    message: ₩
                        knownUsers.get(message.chat.id)    == None, commands=['help'])
def    send_welcome(message):
    bot.send_message(message.chat.id,    allMessage.commands['help_AfterAuthorized'])

@bot.message_handler(func=lambda    message: ₩
```

```
                        knownUsers.get(message.chat.id).userStep == 2, commands=['help'])
def send_welcome(message):
    bot.send_message(message.chat.id, allMessage.commands['help_BeforeAuthorized'])


#
#
#User information:
@bot.message_handler(func=lambda message: True, commands=['information'])
def informaiton_user(message):
    data = 'clientid : ' + knownUsers.get(message.chat.id).clientid + '₩n' +₩
           'remoteuser : ' + knownUsers.get(message.chat.id).remoteUser + '₩n' +₩
           'remotehost : ' + knownUsers.get(message.chat.id).remoteHost

    bot.send_message(message.chat.id, data)

#
#
#Activate, and deactivation bot.
@bot.message_handler(func=lambda message: True, commands=['on'])
def activate_user(message):
    if knownUsers.get(message.chat.id) == None:
        newUser = User(message.chat.id)
        newUser.userName = ''
        newUser.sshUser = config.sshUser
        newUser.sshPassword = config.sshPassword
        newUser.sshHost = config.sshHost
        newUser.cdCommand = config.sshHomeDirectory
        newUser.userStep = 1
        newUser.remoteUser = ''
        newUser.remotePassword = ''
        newUser.remoteHost = ''
        newUser.clientid = ''
        knownUsers[message.chat.id] = newUser

        password = bot.send_message(message.chat.id, "봇 비밀번호를 입력해주세요 :")
          bot.register_next_step_handler(password, check_password)
    elif knownUsers.get(message.chat.id).userStep == 2:
        bot.send_message(message.chat.id, "현재 봇에 권한이 연결된 상태입니다.")

def check_password(message):
    if message.text == config.botPassword:
          knownUsers.get(message.chat.id).userStep = 2
          knownUsers.get(message.chat.id).userName = message.chat.first_name

        bot.send_message(message.chat.id, "접속을 환영합니다!")
    else:
        password = bot.send_message(message.chat.id, "패스워드가 틀렸습니다. 다시
입력해주세요 :")
          bot.register_next_step_handler(password, check_password)


@bot.message_handler(func=lambda message: ₩
                        knownUsers.get(message.chat.id).userStep == 2, commands=['off'])
```

```
def    deactivate_user(message):
    knownUsers.pop(message.chat.id,  None)
    bot.send_message(message.chat.id,  "연결을    해제하였습니다")




#
#
#client    id
@bot.message_handler(func=lambda    message: ₩
                        knownUsers.get(message.chat.id).userStep          ==          2,
commands=['clientid'])
def    request_client_id(message):
    knownUsers.get(message.chat.id).userStep    = 1
    clientid =    bot.send_message(message.chat.id, "점검을 신청한 회원 id    입력 :")
    bot.register_next_step_handler(clientid,    add_client_id)

def add_client_id(message):
    f =    open("/joongbu_script/jb/userinfo",  'w+')
    knownUsers.get(message.chat.id).userStep    = 2
    knownUsers.get(message.chat.id).clientid    = message.text
    sqlconn                                                                                    =
pymysql.connect(host='127.0.0.1',user='nadau',passwd='8648',db='joongbu_db',charset='utf8')
    cur = sqlconn.cursor()
    cur.execute("select id from user    where id like '{0}'".format(message.text))
    result = cur.fetchone()

    if result == None:
        bot.send_message(message.chat.id,    "아이디가 존재하지 않습니다.")
        bot.send_message(message.chat.id,    "/clientid")
    else:
        bot.send_message(message.chat.id,    "아이디가 존재합니다.")
        re = ''.join(filter(str.isalnum,    result))
        f.write(re)
        f.close()
        bot.send_message(message.chat.id,    "/remoteuser")




#
#
#Send    user,passwd
@bot.message_handler(func=lambda    message: ₩
                        knownUsers.get(message.chat.id).userStep          ==          2,
commands=['remoteuser'])
def    request_send_user(message):
    knownUsers.get(message.chat.id).userStep    = 1
    remoteUser =    bot.send_message(message.chat.id, "전송할 서버 user 입력 :")
      bot.register_next_step_handler(remoteUser,  add_send_user)
def    add_send_user(message):
    if ((config.rootPermission)  or    (message.text  !=  'root')):
            knownUsers.get(message.chat.id).remoteUser  =  message.text
```

```
        remotePassword = bot.send_message(message.chat.id, "전송할 서버 비밀번호 입력
:")
            bot.register_next_step_handler(remotePassword, add_send_password)
    else:
        bot.send_message(message.chat.id, "Root not allowed")

def add_send_password(message):
    knownUsers.get(message.chat.id).userStep = 2
      knownUsers.get(message.chat.id).remotePassword = message.text
    bot.send_message(message.chat.id, "/remotehost")

#
#
#send host connect
@bot.message_handler(func=lambda message: ₩
                        knownUsers.get(message.chat.id).userStep == 2,
commands=['remotehost'])
def request_ssh_host(message):
    knownUsers.get(message.chat.id).userStep = 1
    sendDomain = bot.send_message(message.chat.id, "전송할 도메인(IP)주소 입력 :")
      bot.register_next_step_handler(sendDomain, add_send_host)

def add_send_host(message):
    f = open("/joongbu_script/jb/userinfo", 'a')
    knownUsers.get(message.chat.id).userStep = 2
    knownUsers.get(message.chat.id).remoteHost = message.text
    f.write('₩n')
      f.write(knownUsers.get(message.chat.id).remoteHost)
    f.close()
    bot.send_message(message.chat.id, "/sendfile")




#
#
#paramiko file send
@bot.message_handler(func=lambda message: ₩
                        knownUsers.get(message.chat.id).userStep == 2,
commands=['sendfile'])
def send_file(message):
        try:
            client = paramiko.SSHClient()
              client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
              client.connect(hostname=knownUsers.get(message.chat.id).remoteHost,
username=knownUsers.get(message.chat.id).remoteUser,
password=knownUsers.get(message.chat.id).remotePassword)
              scp=SCPClient(client.get_transport())
            scp.put('/joongbu_script/jb', '/', recursive=True )
            scp.close()
            bot.send_message(message.chat.id, "전송 완료")
            bot.send_message(message.chat.id, "/serverscan")
```

```
        except paramiko.AuthenticationException:
            bot.send_message(message.chat.id,    "ID 혹은 비밀번호가 틀렸습니다")
            bot.send_message(message.chat.id,    "/remoteuser")

        except    paramiko.ssh_exception.NoValidConnectionsError:
            bot.send_message(message.chat.id,    "IP주소가 틀렸습니다")
            bot.send_message(message.chat.id,    "/remotehost")

#
#server    scan
@bot.message_handler(func=lambda    message: ₩
                        knownUsers.get(message.chat.id).userStep          ==          2,
commands=['serverscan'])
def    send_file(message):
        client = paramiko.SSHClient()
          client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
          client.connect(hostname=knownUsers.get(message.chat.id).remoteHost,
username=knownUsers.get(message.chat.id).remoteUser,
password=knownUsers.get(message.chat.id).remotePassword)
        stdin, stdout, stderr =    client.exec_command("sh /jb/check.sh")
        data = stdout.read() + stderr.read()
        bot.send_message(message.chat.id,    "검사완료")
        client.close()




#
#
#Set    ssh user and ssh password
@bot.message_handler(func=lambda    message: ₩
                        knownUsers.get(message.chat.id).userStep          ==          2,
commands=['setsshuser'])
def    request_ssh_user(message):
    knownUsers.get(message.chat.id).userStep    = 1
    sshUser = bot.send_message(message.chat.id,    "ssh user 입력 :")
    bot.register_next_step_handler(sshUser,    add_ssh_user)

def    add_ssh_user(message):
    if ((config.rootPermission) or    (message.text != 'root')):
            knownUsers.get(message.chat.id).sshUser = message.text
        sshPassword =    bot.send_message(message.chat.id, "ssh 비밀번호 입력    :")
          bot.register_next_step_handler(sshPassword, add_ssh_password)
    else:
        bot.send_message(message.chat.id,    "Root not allowed")

def    add_ssh_password(message):
    knownUsers.get(message.chat.id).userStep    = 2
      knownUsers.get(message.chat.id).sshPassword = message.text



#
#
#Set    host(IP) for ssh connection:
```

```
@bot.message_handler(func=lambda message: ₩
                     knownUsers.get(message.chat.id).userStep == 2,
commands=['setsshhost'])
def request_ssh_host(message):
    knownUsers.get(message.chat.id).userStep = 1
    sshDomain = bot.send_message(message.chat.id, "접속할 도메인(IP)주소 입력 :")
    bot.register_next_step_handler(sshDomain, add_ssh_host)

def add_ssh_host(message):
    knownUsers.get(message.chat.id).userStep = 2
    knownUsers.get(message.chat.id).sshHost = message.text

#
#
#Check exist sshUser:
@bot.message_handler(func=lambda message: ₩
                     ((knownUsers.get(message.chat.id).userStep == 2) and ₩
                       (knownUsers.get(message.chat.id).sshUser == '') and ₩
                       (knownUsers.get(message.chat.id).sshPassword == '')), ₩
                       content_types=["text"])
def ssh_user_not_exist(message):
    bot.send_message(message.chat.id, "ssh 유저가 없습니다. /setsshuser 혹은 /help 입력")

#
#
#Check exist sshHost:
@bot.message_handler(func=lambda message: ₩
                       ((knownUsers.get(message.chat.id).userStep == 2) and ₩
                     (knownUsers.get(message.chat.id).sshHost == '')), ₩
                       content_types=["text"])
def ssh_user_not_exist(message):
    bot.send_message(message.chat.id, "Ssh host not exist. Use /setsshhost or /help")

#
#
#Do ssh command:
@bot.message_handler(func=lambda message: ₩
                     knownUsers.get(message.chat.id).userStep == 2,
content_types=["text"])
def do_ssh_command(message):
        #check and remember 'cd'-command
        if (message.text[0:3] == 'cd '):
            if knownUsers.get(message.chat.id).cdCommand[-1] != '/':
                knownUsers.get(message.chat.id).cdCommand += '/'
            if message.text[3] == '/':
                knownUsers.get(message.chat.id).cdCommand = message.text.split()[1]
            elif message.text[3:5] == '..':
                flag = knownUsers.get(message.chat.id).cdCommand.split('/')
                knownUsers.get(message.chat.id).cdCommand = '/'
                i = 0
                for element in flag:
                    if ((i == 0) or (i == len(flag)-2) or (i == len(flag)-1)):
                        i += 1
                        continue
```

```
                knownUsers.get(message.chat.id).cdCommand += element + '/'
            i += 1
        else:
            bot.send_message(message.chat.id, message.text[3:5])
        if knownUsers.get(message.chat.id).cdCommand[-1] =='/':
            knownUsers.get(message.chat.id).cdCommand += message.text.split()[1]
        else:
            knownUsers.get(message.chat.id).cdCommand += '/' + message.text.split()[1]
    #ban 'vi', 'less', 'more', 'tail -f':
    elif ((message.text[0:3] == 'vi ') or (message.text[0:5] == 'less ') or ₩
        (message.text[0:5] == 'more ') or (message.text[0:8] == 'tail -f ')):
            bot.send_message(message.chat.id, "Please don't use dynamic command")
    #Do ssh command:
    else:
        try:
            client = paramiko.SSHClient()
            client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
            client.connect(hostname=knownUsers.get(message.chat.id).sshHost, ₩
                    username=knownUsers.get(message.chat.id).sshUser,
                    password=knownUsers.get(message.chat.id).sshPassword)
            sshCommand = 'cd ' +knownUsers.get(message.chat.id).cdCommand + '; ' + ₩
                    message.text
            stdin, stdout, stderr = client.exec_command(sshCommand)
            data = stdout.read() + stderr.read()
            if (sys.getsizeof(data) < 34):
                pass
            elif (sys.getsizeof(data) > 3000):
                bot.send_message(message.chat.id, ₩
                    "Answer from server is too large")
            else:
                bot.send_message(message.chat.id, data)
            client.close()
        except Exception as e:
            bot.send_message(message.chat.id, e)

        #log user actions:
        if config.logging:
            logFile = open(config.logFile, 'a')
            logFile.write("User chat id: '" + str(knownUsers.get(message.chat.id).userChatId) + ₩
                "'; User name: '" + str(knownUsers.get(message.chat.id).userName) + ₩
                "'; Command: '" + str(sshCommand) + "'₩n")
            logFile.close()



while True:
    if __name__ == '__main__':
        bot.polling(none_stop=True)
```

```
    time.sleep(10)
```

텔레그램 도움말 메시지 : allmessage.py

```
# -*-    coding: utf-8 -*-


commands    = {
                'start_AfterAuthorized':
                    u'안녕하세요 1석2조 봇 입니다.₩n₩n'
                    u'봇과 연결을 원하시면 /on 입력₩n',

                'start_BeforeAuthorized':
                    u'안녕하세요 1석2조 봇입니다.₩n'
                    u'봇의 정보를 알고싶으시면 /information 입력₩n'
                    u'다른 명령어들을 보고싶으시면 /help 입력',

                'help_AfterAuthorized':
                    u'안녕하세요 1석2조 봇 입니다.₩n₩n'
                    u'봇과 연결을 원하시면 /on 입력₩n',

                'help_BeforeAuthorized':
                    u'명령어:₩n'
                    u'/off - 봇과 연결을 종료합니다.₩n'
                    u'/information - 설정한 ssh 정보 출력₩n'
                    u'/clientid - 서비스 신청한    회원 id입력₩n'
                    u'/remoteuser - 파일전송 할 ID 및 비밀번호 설정₩n'
                    u'/remotehost - 파일전송 할 IP설정₩n'
                    u'/sendfile - 파일 전송₩n'
                    u'/serverscan - 서버    점검 시작₩n',

}
```

텔레그램 설정파일 : config.py

```
# -*-    coding: utf-8 -*-

#Telegram    Bot API token (ask BotFather):
botToken    = '1771478021:AAEeOXPZXvYp-sqp0EcxhofU2YslQxUSAAo'

#Password    for authentication in bot:
botPassword    = '1234'

#Default    data for ssh-connection:
```

```
sshUser     = 'root'
sshPassword     =
sshHost     = '192.168.111.100'
sshHomeDirectory     = '/joongbu_script'

#Root     permission. If 'True" - root allowed
rootPermission     = True

#Logging.     If 'True" - logging ON
logging     = True
logFile     = '/joongbu_script/telegram/telegramBot.log'
```

5.2 발표 자료

CONTENTS

01 조원편성

02 주제선정

03 구상도

04 개발 일정 및 진행 사항

01

조원 편성

01. 조원 편성

이름	역 할
남승택	조장, 취약점 진단 스크립트, telegram-bot 개발
박세빈	취약점 진단 스크립트, telegram-bot 개발
신자연	취약점 진단 스크립트, telegram-bot 개발
왕나원	취약점 진단 스크립트, front-end
양재희	취약점 진단 스크립트, front-end

02
주제 선정

02. 주제선정

서버 취약점 진단을 선정하는 이유

- [정보통신기반 보호법] 제 9조에 따라, 주요정보통신기반시설 관리기관은 매년 취약점 분석 평가를 실시하여야 한다.
- 사소해 보이는 취약점을 찾아서 위협 요소에 대한 조치 방안과 보호 대책을 제시하기 때문에 예방을 할 수 있다.

02. 주제선정

텔레그램 봇 선정 이유

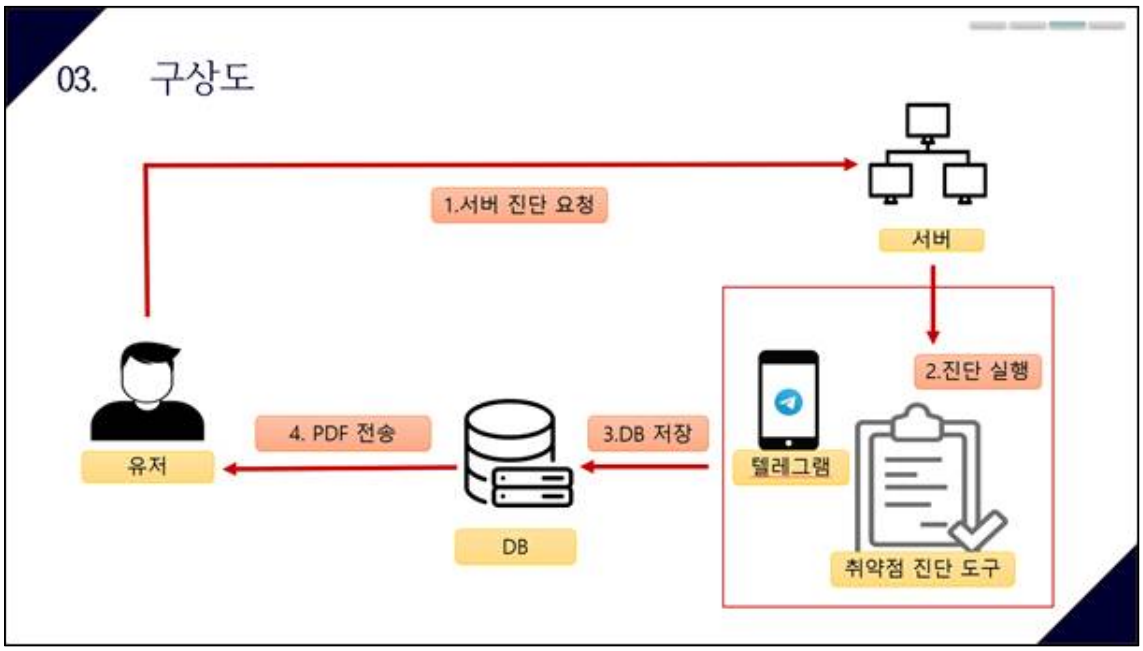
- Python 언어로 개발 가능
-> 사용법이 간단, 친숙한 언어
- Python 언어로 개발 가능 API가 공개되어 다른 프로그램 또는 플랫폼과 연계해 사용 가능

python

03

구상도

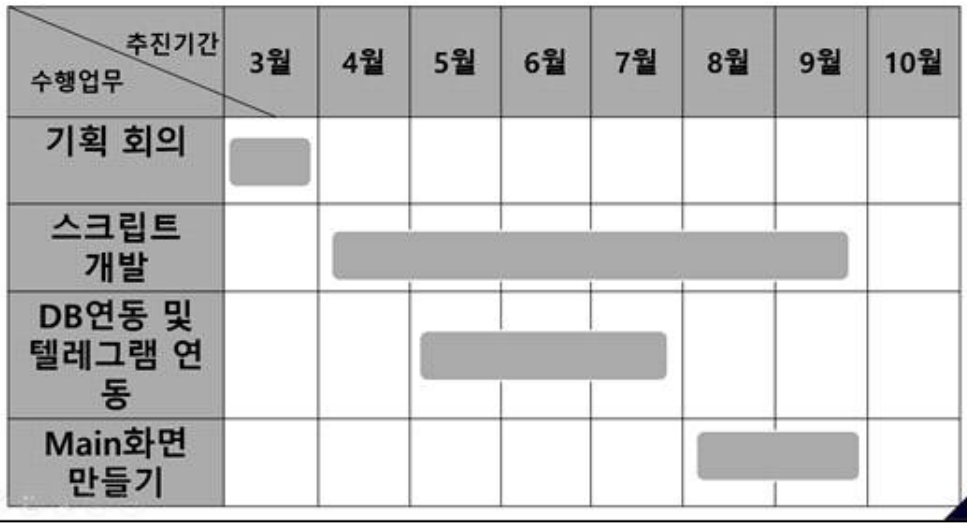
03. 구상도



04

개발 일정 및 진행 사항

04. 개발 일정



계정 관리 15개 항목

04. 진행 사항 (script)

분류	항목	진행사항
계정 관리	Root 계정 원격 접속 제한	●
	패스워드 복잡성 설정	●
	계정 잠금 임계값 설정	●
	패스워드 파일 보호	●
	Root 이외의 UID가 0금지	●
	Root 계정 su 제한	●
	패스워드 최소 길이 설정	●
	패스워드 최대 사용기간 설정	●
	패스워드 최소 사용기간 설정	●
	불필요한 계정 제거	●
	관리자 그룹에 최소한의 계정포함	●
	계정이 존재하지 않는 GID금지	●
	동일한 UID금지	●
	사용자 shell 점검	▲
	Session Timeout 설정	●

파일 및 디렉터리 관리 20개 항목

04. 진행 사항 (scrtpt)

분류	항목	진행사항	항목	진행사항
파일 및 디렉터리 관리	Root 홈, 패스 디렉터리 권한 및 패스 설정	●	World writable 파일 점검	●
	파일 및 디렉터리 소유자 설정	●	/dev에 존재하지 않는 device 파일 점검	●
	/etc/passwd 파일 소유자 및 권한 설정	●	$HOME/.rhosts, hosts.equiv 사용 금지	●
	/etc/shadow 파일 소유자 및 권한 설정	●	접속 IP 및 포트 제한	●
	/etc/hosts 파일 소유자 및 권한 설정	●	Hosts.lpd 파일 소유자 및 권한 설정	●
	/etc/(x)inetd.conf 파일 소유자 및 권한 설정	●	NIS 서비스 비활성화	●
	/etc/syslog.conf 파일 소유자 및 권한 설정	●	UMASK 설정 관리	●
	/etc/services 파일 소유자 및 권한 설정	●	홈디렉터리 소유자 및 권한 설정	●
	SUID, SGID, Sticky bit 설정 파일 점검	●	홈디렉터리로 지정한 디렉터리의 존재 관리	●
	사용자, 시스템 시작 파일 및 환경파일 소유자 및 권한 설정	●	숨겨진 파일 및 디렉터리 검색 및 제거	●

04. 진행 사항 (window script)

분류	항목	진행사항	분류	항목	진행사항
계정 관리	Administrator 계정 이름 변경 또는 보안성 강화	●	계정 관리	패스워드 최소 암호 길이	●
	Guest 계정 비활성화	●		패스워드 최대 사용 기간	●
	불필요한 계정 제거	●		패스워드 최소 사용 기간	●
	계정 잠금 임계값 설정	●		마지막 사용자 이름 표시 안 함	●
	해독 가능한 암호화를 사용하여 암호 저장 해제	●		로컬 로그온 허용	●
	관리자 그룹에 최소한의 사용자 포함	●		익명 SID/이름 변환 허용 해제	●
	Everyone 사용권한을 익명 사용자에 적용 해제	●		최근 암호 기억	●
	계정 잠금 기간 설정	●		콘솔 로그온 시 로컬 계정에서 빈 암호 사용 제한	●
	패스워드 복잡성 설정	●		원격터미널 접속 가능한 사용자 그룹 제한	●

04. 진행 사항 (window script)

분류	항목	진행사항	항목	진행사항	항목	진행사항
서비스 관리	공유 권한 및 사용자 그룹 설정	●	IIS 가상 디렉토리 삭제	●	최신 서비스팩 적용	●
	하드디스크 기본 공유 제거	●	IIS 데이터파일 ACL 적용	●	터미널 서비스 암호화 수준 설정	●
	불필요한 서비스 제거	●	IIS 미사용 스크립트 매핑 제거	●	IIS 웹 서비스 정보 숨김	●
	IIS 서비스 구동 점검	●	IISExec 명령어 쉘호출 진단	●	SNMP 서비스 구동 점검	●
	IIS 디렉토리 리스팅 제거	●	IIS WebDAV 비활성화	●	SNMP 서비스 커뮤니티스트링의 복잡성 설정	●
	IIS CGI 실행 제한	●	NetBIOS 바인딩 서비스 구동 점검	●	SNMP Access control 설정	●
	IIS 상위 디렉토리 접근 금지	●	FTP 서비스 구동 점검	●	DNS 서비스 구동 점검	●
	IIS 불필요한 파일 제거	●	FTP 디렉토리 접근권한 설정	●	HTTP/FTP/SMTP 배너 차단	●
	IIS 웹프로세스 권한 제한	●			Telnet 보안 설정	●
	IIS 링크 사용 금지	●	Anonymous FTP 금지	●	불필요한 ODBC/OLE-DB 데이터소스와 드라이브 제거	●
	IIS 파일 업로드 및 다운로드 제한	●	FTP 접근 제어 설정	●	원격터미널 접속 타임아웃 설정	●
	IIS DB 연결 취약점 점검	●	DNS Zone Transfer 설정	●	예약된 작업에 의심스러운 명령이 등록되어 있는지 점검 중	●
			RDS 제거	●		

04. 진행 사항 (window script)

분류	항목	진행사항
패치 관리	최신 HOT FIX 적용	●
	백신 프로그램 업데이트	●
	정책에 따른 시스템 로깅설정	●
로그 관리	로그의 정기적 검토 및 보고	●
	원격으로 액세스 할 수 있는 레지스트리 경로	●
	이벤트 로그 관리 설정	●
	원격에서 이벤트 로그파일 접근 차단	●

04. 진행 사항 (window script)

분류	항목	진행사항	항목	진행사항
서비스 관리	백신 프로그램 설치	●	Dos 공격 방어 레지스트리 설정	●
	SAM 파일 접근 통제 설정	●	사용자가 프린터 드라이버를 설치할 수 없게 함	●
	화면보호기 설정	●	세션 연결을 중단하기 전에 필요한 유휴시간	●
	로그온 하지 않고 시스템 종료 허용 해제	●	경고 메시지 설정	●
	원격 시스템에서 강제로 시스템 종료	●	사용자별 홈 디렉토리 권한 설정	●
	보안감사를 로그할 수 없는 경우 즉시 시스템 종료 해제	●	LAN Manager 인증 수준	●
	SAM 계정과 공유의 익명 열거 허용 안함	●	보안 채널 데이터 디지털 암호화 또는 서명	●
	Autologon 기능 제어	●	파일 및 디렉토리 보호	●
	이동식 미디어 포맷 및 꺼내기 허용	●	컴퓨터 계정 암호 최대 사용 기간	●
	디스크 볼륨 암호화 설정	●	시작 프로그램 목록 분석	●

04. 진행 사항 (script)

Main 코드

```
#!/bin/bash

clear

while [ True ]
do
        echo -e "\n**********취약점 진단 프로그램**********\n"
        echo -e "                                만든이 : 2조(이게뭐조)\n"
       echo -en "1.진단하기\n\n2.진단결과 확인하기\n\n3.만든이 보기\n\n4.나가기\n\n입력:"
       read check


   if [ $check -eq 1 ]; then

      while [ True ]
      do

         echo -n "회원ID 입력:"
         read TMPID

         mysql -h localhost -usebin -p1234 -e "select id from user where id like '$TMPID';" joongbu_db > /joongbu_script/UID.txt

         if [ -z "`cat /joongbu_script/UID.txt`" ]; then
            clear
            echo "존재하지 않는 아이디 입니다."
            continue
         else
            clear
            echo "아이디가 존재 합니다."
            uid=`cat /joongbu_script/UID.txt | tail -1`
            #rm -rf /joongbu_script/UID.txt
            break
         fi
      done
```

04. 진행 사항 (script)

Main 코드

```
< 중부대학교 고양캠퍼스 소속 >

91613703 남승택                    91812335 박세빈

91812608 신자연                    91812701 왕나원

91812672 양재희

< 주요 프로젝트 내용 >

리눅스 서버 구축 및 취약점 진단 스크립트 작성

윈도우 서버 구축 및 취약점 진단 스크립트 작성

진단 결과 pdf 파일로 확인

DB 연동 및 텔레그램 연동

메뉴로 돌아가시겠습니까? (y):
```

```
**********취약점 진단 프로그램**********

                                만든이 : 2조(이게뭐조)

1. 진단하기

2. 진단결과 확인하기

3. 만든이 보기

4. 나가기

입력:
```

04. 진행 사항 (script)

Main 코드

```
원격지 ID입력 : root
원격지 IP 입력 :192.168.111.100
```

```
총 합 : 73
전체 양 호 : 46
전체 취 약 : 22
자체 검 사 : 2
전체 검사 불가 : 3
Connection to 192.168.111.100 closed.
 [3: ]
```

```
아이디가 존재 합니다.

OS 종류를 선택 하세요. (1. linux(centos) 2. windows server 2012) :
```

```
••••••••••••••••••••취약점 진단 프로그램••••••••••••••••••••

                                        만든이 : 2조(이게뭐조)

1. 진단하기

2. 진단결과 확인하기

3. 만든이 보기

4. 나가기

입력:
```

```
〈 U-53 사용자 shell 점검 〉

상태 : 취약
로그인이 필요하지 않은 계정에 /bin/false(/sbin/nologin)쉘이 부여되지 않았습니다.

================================================================================

〈 U-54 Session Timeout 설정 〉

상태 : 취약
TMOUT이 설정되어 있지 않습니다

================================================================================

                                        계정관리 총 합 : 15
                                        계정관리 양 호 : 5
                                        계정관리 취 약 : 10
                                        계정관리 검사 불가 : 0
```

04. 진행 사항 (script)

Main 코드

```
••••••••••••••••••••취약점 진단 프로그램••••••••••••••••••••

                                        만든이 : 2조(이게뭐조)

1. 진단하기

2. 진단결과 확인하기

3. 만든이 보기

4. 나가기

입력:2
```

```
•••••••••••••••• 계정관리 부분 ••••••••••••••••
                계정관리 양 호 : 5
                계정관리 취 약 : 10
                계정관리 검사 불가 : 0
                계정 관리 총 합 : 15

•••••••••••••••• 파일 및 디렉터리 관리••••••••••••••••
                파일및 디렉터리 관리 양 호 : 12
                파일및 디렉터리 관리 취 약 : 5
                파일및 디렉터리 관리 검사 불가 : 2
                파일및 디렉터리 관리 자체 검사(U-60) : 1
                파일 및 디렉터리 관리 총 합 : 20

•••••••••••••••• 서비스 관리 ••••••••••••••••
                서비스 관리 양 호 : 28
                서비스 관리 취 약 : 6
                서비스 관리 검사 불가 : 1
                서비스 관리 총 합 : 35

•••••••••••••••• 패치 및 로그 관리 ••••••••••••••••
                패치 및 로그 관리 양 호 : 1
                패치 및 로그 관리 취 약 : 1
                패치 및 로그 관리 검사 불가 : 0
                패치 및 로그 관리 자체 검사(U-43) : 1
                패치 및 로그 관리 총 합 : 3

총 합 : 73
전체 양 호 : 46
전체 취 약 : 22
자체 검 사 : 2
전체 검사 불가 : 3

메뉴로 돌아가시겠습니까? (y):
```

04. 진행 사항 (script)

계정 관리 U-03

```
#!/bin/bash

echo ""
echo "< U-03 계정 잠금 임계값  설정 >"
echo ""

if [ -f "/etc/pam.d/system-auth" ];
then
        grep pam_tally.so /etc/pam.d/system-auth > /dev/null
        if [ $? -eq 0 ];
        then
                echo -e "상태 : \e[1;36m양호\e[0m"
                echo "계정관리,U-03,계정 잠금 임계값 설정,Safety,$userid,$serverip,$DATE,1 " >> /jb/result.txt
                AccountSafety=`expr $AccountSafety + 1`

        else
                echo -e "상태 : \e[1;31m취약\e[0m"
                echo "계정 잠금 임계값이 설정되어 있지 않거나, 5 이하의 값으로 설정되어 있지 않습니다."
                echo "계정관리,U-03,계정 잠금 임계값 설정,Risk,$userid,$serverip,$DATE,0,계정 잠금 임계값을 5이하로 설정하세요." >> /jb/result.txt
                AccountRisk=`expr $AccountRisk + 1`
        fi
else
        echo -e "상태 : \e[1;33m검사 불가\e[0m"
        echo "/etc/pam.d/system-auth 파일이 존재하지 않습니다."
        echo "계정관리,U-03,계정 잠금 임계값 설정,Risk,$userid,$serverip,$DATE,0,파일이 없습니다." >> /jb/result.txt
        AccountError=`expr $AccountError + 1`
fi
echo""
echo ================================================================================
```

04. 진행 사항 (script)

서비스 관리 U-32

```
#!/bin/bash
echo ""
echo "< U-32 일반사용자의 Sendmail 실행 방지 >"
echo ""

ps -ef | grep sendmail | grep -v grep > /jb/ee.txt

if [ -s "/jb/ee.txt" ]; then
        grep -v '^ *#' /etc/mail/sendmail.cf | grep PrivacyOptions | grep restrictqrun  > /jb/ff.txt
        if [ -s "/jb/ff.txt" ]; then
                echo -e "상태 : \e[1;36m양호\e[0m"
                echo "서비스관리,U-32,일반사용자의 Sendmail 실행 방지,Safety,$userid,$serverip,$DATE,1" >> /jb/result.txt
                ServiceSafety=`expr $ServiceSafety + 1`

        else
                echo -e "상태 : \e[1;31m취약\e[0m"
                echo "SMTP서비스를 사용하는데 일반사용자의 Sendmail 실행 방지가 설정되어 있지 않습니다."
                echo "서비스관리,U-32,일반사용자의 Sendmail 실행 방지,Risk,$userid,$serverip,$DATE,0,/etc/mail/sendmail.cf파일에서 restrictqrun옵션을 추가하세요." >> /jb/result.txt
                ServiceRisk=`expr $ServiceRisk + 1`

        fi
else
        echo -e "상태 : \e[1;36m양호\e[0m"
        echo "서비스관리,U-32,스팸 메일 릴레이 제한,Safety,$userid,$serverip,$DATE,1" >> /jb/result.txt
        ServiceSafety=`expr $ServiceSafety + 1`

fi

echo ""
echo ================================================================================

rm -rf /jb/ee.txt
rm -rf /jb/ff.txt
```

04. 진행 사항 (script)

계정 관리 U-46

```
#!/bin/bash

echo ""
echo "< U-60 숨겨진 파일 및 디렉터리 검색 및 제거>"
echo ""


#c_time=`ls -l --time-style full-iso /root/anaconda-ks.cfg | awk '{print $6}'`
#find /home -type f -name ".*" -newermt '2020-12-21'

#find /home -name ".*" -type f -newer /root/anaconda-ks.cfg

scan_dir=("/boot" "/usr" "/bin" "/sbin" "/etc/" "/tmp" "/home" "/var")

for i in ${scan_dir[@]}
do
        echo ------$i 숨김파일들--------- >> /jb/tmp.txt
        find $i -name ".*" -type f -newer /root/anaconda-ks.cfg >> /jb/tmp.txt
        echo "" >> /jb/tmp.txt
done


echo "상태 : N/A"
echo ""
cat /jb/tmp.txt
echo "파일 및 디렉터리 관리,U-60,숨겨진 파일 및 디렉터리 검색 및 제거,N/A,$userid,$serverip,$DATE,1" >> /jb/result.txt

echo ""
echo "================================================================================"
rm -rf /jb/tmp.txt
```

04. 진행 사항 (script)

파일 및 디렉터리 관리 U-60

```
#!/bin/bash

echo ""
echo "< U-60 숨겨진 파일 및 디렉터리 검색 및 제거>"
echo ""


#c_time=`ls -l --time-style full-iso /root/anaconda-k
#find /home -type f -name ".*" -newermt '2020-12-21'

#find /home -name ".*" -type f -newer /root/anaconda-

scan_dir=("/boot" "/usr" "/bin" "/sbin" "/etc/" "/tmp

for i in ${scan_dir[@]}
do
        echo ------$i 숨김파일들--------- >> /jb/tmp.
        find $i -name ".*" -type f -newer /root/anaco
        echo "" >> /jb/tmp.txt
done


echo "상태 : N/A"
echo ""
cat /jb/tmp.txt
echo "파일 및 디렉터리 관리,U-60,숨겨진 파일 및 디렉터

echo ""
echo "================================================
rm -rf /jb/tmp.txt
```

```
< U-60 숨겨진 파일 및 디렉터리 검색 및 제거>
상태 : N/A

------/boot 숨김파일들----------

------/usr 숨김파일들----------
/usr/lib64/firefox/fonts/.uuid

------/bin 숨김파일들----------

------/sbin 숨김파일들----------

------/etc/ 숨김파일들----------

------/tmp 숨김파일들----------
/tmp/.X0-lock
/tmp/.X1024-lock

------/home 숨김파일들----------
/home/hdfs/.esd_auth
/home/hdfs/.bash_history
/home/hdfs/.vim/.netrwhist
/home/hdfs/.viminfo
/home/yarn/.esd_auth
/home/yarn/.bash_history

------/var 숨김파일들----------
/var/lib/sss/secrets/.secrets.mkey
/var/lib/gdm/.ICEauthority

================================================
```

04.진행 사항 (Windows 10)

계정 관리 W-5

```
@ECHO OFF
ECHO.
ECHO ^< W-5 해독 가능한 암호화를 사용하여 암호 저장 기준 ^>
ECHO.

secedit /export /cfg C:₩jb₩LocalSecurityPolicy.txt | find /v "작업을" | find /v "다시한"

TYPE C:₩jb₩LocalSecurityPolicy.txt | find /i "ClearTextPassword" | find "0" > NUL

if not errorlevel 1 echo "결과 : 양호"
if not errorlevel 1 echo "계정 관리,W-05,해독 가능한 암호화를 사용하여 암호 저장기준,Safety,%userid%,%serverip%,%date%,1" >> C:₩Users₩%name%₩jb₩result.txt
if not errorlevel 1 set /a AccounSafety=%AccountSafety%+1

if errorlevel 1 echo "결과 : 취약"&echo."해독 가능한 암호화를 사용하여 암호 저장 정책이 사용으로 되어 있습니다."
if errorlevel 1 echo "계정 관리,W-05,해독 가능한 암호화를 사용하여 암호 저장기준,Risk,%userid%,%serverip%,%date%,0,해독 가능한 암호화를 사용하여 암호 저장 정책을 사용 안 함으로 설정하세요." >> C:₩Users₩%name%₩jb₩result.txt
if errorlevel 1 set /a AccountRisk=%AccountRisk%+1

ECHO.
ECHO.********************************************************************************
```

```
< W-5 해독 가능한 암호화를 사용하여 암호 저장 기준 >

"결과 : 양호"
지정된 경로를 찾을 수 없습니다.

================================================================================
계속하려면 아무 키나 누르십시오 . . .
```

04.진행 사항 (Windows 10)

서비스 관리 W-15

```
@ECHO OFF

ECHO.
ECHO ^< W-17 IIS 파일 업로드 및 다운로드 제한 ^>
ECHO.

dir | find "web.config" || dir | find "applicationHost.config" > C:₩w17.txt
IF %errorlevel% EQU 0 (echo "상태 : 양호"
        echo "서비스 관리,W-17,IIS 파일 업로드 및 다운로드 제한,Safety,%userid%,%serverip%,%date%,1" >> C:₩Users₩%name%₩jb₩result.txt
        set /a ServiceSafety=%ServiceSafety%+1
) ELSE (echo "상태 : 취약"&echo."웹 프로세스의 서버 자원 관리가 되어있지 않습니다."
echo "서비스 관리,W-17,IIS 파일 업로드 및 다운로드 제한,Risk,%userid%,%serverip%,%date%,0,웹 프로세스의 서버 자원 관리를 설정하세요." >> C:₩Users₩%name%₩jb₩result.txt
set /a ServiceRisk=%ServiceRisk%+1
)

del C:₩w17.txt
ECHO.
ECHO.********************************************************************************
```

```
< W-17 IIS 파일 업로드 및 다운로드 제한 >

"상태 : 취약"
"웹 프로세스의 서버 자원 관리가 되어있지 않습니다."
지정된 경로를 찾을 수 없습니다.

================================================================================
계속하려면 아무 키나 누르십시오 . . .
```

04.진행 사항 (Windows 10)

계정 관리 W-49

```
@ECHO OFF
ECHO.
ECHO ^< W-49 패스워드 최소 암호 길이 ^>
ECHO.

secedit /export /cfg C:₩LocalSecurityPolicy.txt

TYPE C:₩LocalSecurityPolicy.txt | find /i "MinimumPasswordLength = " > C:₩passwd.txt

FOR /f "tokens=3" %%a IN (C:₩passwd.txt) DO SET pass_length=%%a

IF %pass_length% GEQ 8 ECHO "상태 : 양호"
IF %pass_length% GEQ 8 ECHO "계정 관리,W-49, 패스워드 최소 암호 길이,Safety,%userid%,%serverip%,%date%,1" >> C:₩Users₩%name%₩jb₩result.txt
IF %pass_length% GEQ 8 set /a AccountSafety=%AccountSafety%+1

IF NOT %pass_length% GEQ 8 ECHO "상태 : 취약"&ECHO."최소 암호 길이가 설정되지 않았거나 8문자 미만으로 설정되어 있는 경우입니다."
IF NOT %pass_length% GEQ 8 ECHO "계정 관리,W-49,패스워드 최소 암호 길이,Risk,%userid%,%serverip%,%date%,0,최소 암호 길이를 8문자 이상으로 설정해주세요." >> C:₩Users₩%name%₩jb₩result.txt
IF NOT %pass_length% GEQ 8 set /a AccountRisk=%AccountRisk%+1

ECHO.
ECHO.==========

del C:₩LocalSecurityPolicy.txt
DEL C:₩passwd.txt
```

```
< W-49 패스워드 최소 암호 길이 >

작업을 성공적으로 완료했습니다.
자세한 정보는 %windir%₩security₩logs₩scesrv.log를 참조하십시오.
"상태 : 취약"
"최소 암호 길이가 설정되지 않았거나 8문자 미만으로 설정되어 있는 경우입니다."
지정된 경로를 찾을 수 없습니다.

계속하려면 아무 키나 누르십시오 . . .
```

04.진행 사항 (Windows 10)

보안 관리 W-76

```
@ECHO OFF
ECHO.
ECHO ^< W-76 사용자별 홈 디렉토리 권한 설정 ^>
ECHO.


dir %systemroot%₩..₩Users | find /v "." | find /v "Public" | find "<DIR>" > c:₩user.txt
TYPE c:₩user.txt | find /i "Everyone" > nul

IF %ERRORLEVEL%==0 (
        ECHO "결과 : 취약"
        ECHO "홈 디렉토리에 Everyone 권한이 있습니다."
        ECHO "보안 관리,W-76,사용자별 홈 디렉토리 권한 설정,Risk,%userid%,%serverip%,%date%,0,홈 디렉토리에 Everyone 권한이 있습니다." >> c:₩Users₩%name%₩jb₩result.txt
        set /a SecurityRisk=%SecurityRisk%+1
) ELSE (
        ECHO "결과 : 양호"
        ECHO "보안 관리,W-76,사용자별 홈 디렉토리 권한 설정,Safety,%userid%,%serverip%,%date%,1" >> c:₩Users₩%name%₩jb₩result.txt
        set /a SecuritySafety=%SecuritySafety%+1
)


ECHO.
ECHO.==========

DEL c:₩user.txt
```

```
< W-76 사용자별 홈 디렉토리 권한 설정 >

"결과 : 양호"
지정된 경로를 찾을 수 없습니다.

계속하려면 아무 키나 누르십시오 . . .
```

04. 진행 사항 (telegram)

```
#Telegram Bot API token
botToken = '[illegible]'

#Bot password
botPassword = [illegible]

#Default ssh-connection
sshUser = 'root'
sshPassword = '[illegible]'
sshHost = '192.168.111.100'
sshHomeDirectory = '/joongbu_script'

#Root permission.
rootPermission = True

#Logging.
logging = True
logFile = '/joongbu_script/telegram/t
```

```
@bot.message_handler(func=lambda message: True, commands=['on'])
def activate_user(message):
    if knownUsers.get(message.chat.id) == None:
        newUser = User(message.chat.id)
        newUser.userName = ''
        newUser.sshUser = config.sshUser
        newUser.sshPassword = config.sshPassword
        newUser.sshHost = config.sshHost
        newUser.cdCommand = config.sshHomeDirectory
        newUser.userStep = 1
        newUser.remoteUser = ''
        newUser.remotePassword = ''
        newUser.remoteHost = ''
        newUser.clientid = ''
        knownUsers[message.chat.id] = newUser

        password = bot.send_message(message.chat.id, "봇 비밀번호를 입력해주세요 :")
        bot.register_next_step_handler(password, check_password)
    elif knownUsers.get(message.chat.id).userStep == 2:
        bot.send_message(message.chat.id, "현재 봇에 권한이 연결된 상태입니다.")

def check_password(message):
    if message.text == config.botPassword:
        knownUsers.get(message.chat.id).userStep = 2
        knownUsers.get(message.chat.id).userName = message.chat.first_name

        bot.send_message(message.chat.id, "접속을 환영합니다!")
    else:
        password = bot.send_message(message.chat.id, "패스워드가 틀렸습니다. 다시 입력해주세요 :")
```

04. 진행 사항 (telegram)

04. 진행 사항 (telegram)

```
@bot.message_handler(func=lambda message: \
                     knownUsers.get(message.chat.id).userStep == 2, commands=['clientid'])
def request_client_id(message):
    knownUsers.get(message.chat.id).userStep = 1
    clientid = bot.send_message(message.chat.id, "점검을 신청한 회원 id 입력 :")
    bot.register_next_step_handler(clientid, add_client_id)

def add_client_id(message):
    f= open("/joongbu_script/jb/userinfo", 'w+')
    knownUsers.get(message.chat.id).userStep = 2
    knownUsers.get(message.chat.id).clientid = message.text
    sqlconn = pymysql.connect(host='127.0.0.1',user='nadau',passwd='8648',db='joongbu_db',charset='utf8')
    cur = sqlconn.cursor()
    cur.execute("select id from user where id like '{0}'".format(message.text))
    result = cur.fetchone()

    if result == None:
       bot.send_message(message.chat.id, "아이디가 존재하지 않습니다.")
       bot.send_message(message.chat.id, "/clientid")
    else:
       bot.send_message(message.chat.id, "아이디가 존재합니다.")
       re = ''.join(filter(str.isalnum, result))
       f.write(re)
       f.close()
       bot.send_message(message.chat.id, "/remoteuser")
```

04. 진행 사항 (telegram)

```
@bot.message_handler(func=lambda message: \
                     knownUsers.get(message.chat.id).userStep == 2, commands=['clientid'])
def request_client_id(message):
    knownUsers.get(message.chat.id).userStep = 1
    clientid = bot.send_message(message.chat.id, "점검을 신청한 회원 id 입력 :")
```

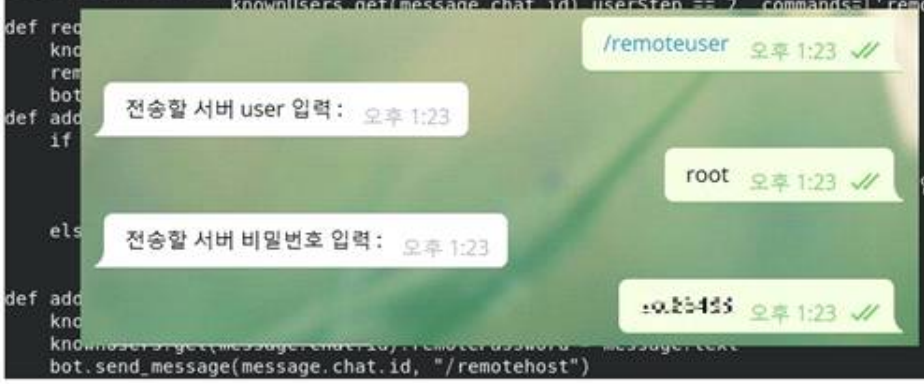

```
       bot.send_message(message.chat.id, "아이디가 존재합니다.")
       re = ''.join(filter(str.isalnum, result))
       f.write(re)
       f.close()
       bot.send_message(message.chat.id, "/remoteuser")
```

04. 진행 사항 (telegram)

```
@bot.message_handler(func=lambda message: \
                     knownUsers.get(message.chat.id).userStep == 2, commands=['remoteuser'])
def request_send_user(message):
    knownUsers.get(message.chat.id).userStep = 1
    remoteUser = bot.send_message(message.chat.id, "전송할 서버 user 입력 :")
    bot.register_next_step_handler(remoteUser, add_send_user)
def add_send_user(message):
    if ((config.rootPermission) or (message.text != 'root')):
        knownUsers.get(message.chat.id).remoteUser = message.text
        remotePassword = bot.send_message(message.chat.id, "전송할 서버 비밀번호 입력 :")
        bot.register_next_step_handler(remotePassword, add_send_password)
    else:
        bot.send_message(message.chat.id, "Root not allowed")

def add_send_password(message):
    knownUsers.get(message.chat.id).userStep = 2
    knownUsers.get(message.chat.id).remotePassword = message.text
    bot.send_message(message.chat.id, "/remotehost")
```

04. 진행 사항 (telegram)



04. 진행 사항 (telegram)

```
@bot.message_handler(func=lambda message: \
                     knownUsers.get(message.chat.id).userStep == 2, commands=['remotehost'])
def request_ssh_host(message):
    knownUsers.get(message.chat.id).userStep = 1
    sendDomain = bot.send_message(message.chat.id, "전송할 도메인(IP)주소 입력 :")
    bot.register_next_step_handler(sendDomain, add_send_host)

def add_send_host(message):
    f = open("/joongbu_script/jb/userinfo", 'a')
    knownUsers.get(message.chat.id).userStep = 2
    knownUsers.get(message.chat.id).remoteHost = message.text
    f.write('\n')
    f.write(knownUsers.get(message.chat.id).remoteHost)
    f.close()
    bot.send_message(message.chat.id, "/sendfile")
```

04. 진행 사항 (telegram)

```
@bot.message_handler(func=lambda message: \
                     knownUsers.get(message.chat.id).userStep == 2, commands=['remotehost'])
def request_ssh_host(message):
    kn
    se
    bo

def ad
    f
    kn
    kn
    f.write('\n')
    f.write(knownUsers.get(message.chat.id).remoteHost)
    f.close()
    bot.send_message(message.chat.id, "/sendfile")
```

04. 진행 사항 (telegram)

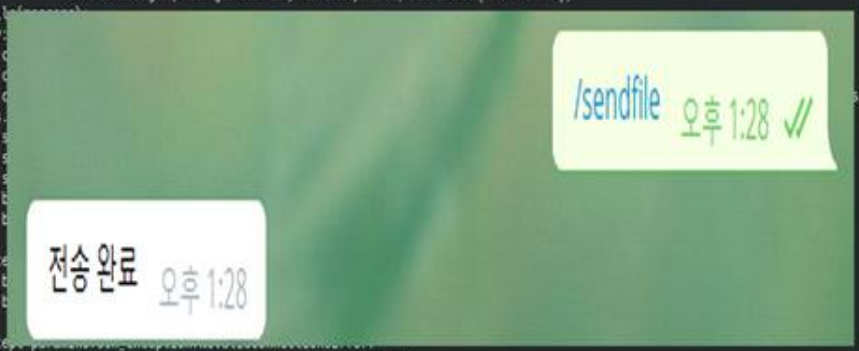
```
@bot.message_handler(func=lambda message: \
                     knownUsers.get(message.chat.id).userStep == 2, commands=['sendfile'])
def send_file(message):
        try:
           client = paramiko.SSHClient()
           client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
           client.connect(hostname=knownUsers.get(message.chat.id).remoteHost, username=knownUsers.get(message.chat.id).remoteUser, password
=knownUsers.get(message.chat.id).remotePassword)
           scp=SCPClient(client.get_transport())
           scp.put('/joongbu_script/jb', '/', recursive=True )
           scp.close()
           bot.send_message(message.chat.id, "전송 완료")
           bot.send_message(message.chat.id, "/serverscan")

        except paramiko.AuthenticationException:
           bot.send_message(message.chat.id, "ID 혹은 비밀번호가 틀렸습니다")
           bot.send_message(message.chat.id, "/remoteuser")

        except paramiko.ssh_exception.NoValidConnectionsError:
           bot.send_message(message.chat.id, "IP주소가 틀렸습니다")
           bot.send_message(message.chat.id, "/remotehost")
```

04. 진행 사항 (telegram)

```
@bot.message_handler(func=lambda message: \
                     knownUsers.get(message.chat.id).userStep == 2, commands=['sendfile'])
def send_fil
        try:
                                                                                    ser, password
=knownUsers.
        exce
        exce
           bot.send_message(message.chat.id, "IP주소가 틀렸습니다")
           bot.send_message(message.chat.id, "/remotehost")
```



04. 진행 사항 (telegram)

```
@bot.message_handler(func=lambda message: \
                knownUsers.get(message.chat.id).userStep == 2, commands=['serverscan'])
def send_file(message):
        client = paramiko.SSHClient()
        client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
        client.connect(hostname=knownUsers.get(message.chat.id).remoteHost, username=knownUsers.get(message.chat.id).remoteUser, password=kn
ownUsers.get(message.chat.id).remotePassword)
        stdin, stdout, stderr = client.exec_command("sh /jb/check.sh")
        data = stdout.read() + stderr.read()
        bot.send_message(message.chat.id, "검사완료")
        client.close()
```

04. 진행 사항 (telegram)

```
@bot.message_handler(func=lambda message: \
                knownUsers.get(message.chat.id).userStep == 2, commands=['serverscan'])
def sen
                                                                                                                    er, password=kn
ownUser
        client.close()
```


Thank you