서버 취약점 자동진단 프로그램 개발

팀 명 : 모그로그

지 도 교 수 : 양환석 교수님

팀 장 : 강성현

팀 원 : 김고운

송소연

윤지예

정지수

2023. 10. 20

중부대학교 정보보호학과

목 차

1. 서 론

1.1 연구배경

현재 대한민국의 수많은 기업과 국가시설 또는 개인이 각각의 서버를 구축하여 운영하고 있다. 하지만 정보통신기술의 발전에 따른 시스템 보안은 중요한 이슈로 부각되고 있으며, 이에 대한 효율적, 효과적인 대응이 필수적으로 요구되고 있다. 공격자들은 시스템 파괴 및 개인정보 유출, 사용자 위장을 통한 침입 등의 피해를 야기하고 그로인해 정보시스템을 운영하는 기관과 개인이 운영하는 서버가 많은 손실을 입고 있다. 이에 따라 KISA(한국인터넷진흥원)에서 제공하는 취약점 진단 가이드를 참고하여 프로그램을 제작하였다.

1.2 연구 필요성

“정보통신기반 보호법” 제 9조에 따라 정기적으로 소관 주요정보통신기반시설의 취약점을 분석 및 평가하여야 하고 취약점에 따른 공격에 노출되지 않도록 할 책임이 있다. 또한 나날이 발전하고 있는 기술에 따른 새로운 취약점에도 재빠른 대비가 필요하며 보이지 않는 잠재적인 위협에도 최소한의 손실을 유발함과 동시에 이를 예방 해야한다. 따라서 시스템 특성에 따른 효과적인 취약점 진단 및 빈틈없는 점검이 필요하다.

1.3 연구 목적 및 주제선정

본 연구는 기업과 국가기관 또는 개인이 주로 사용하는 서버의 형태를 조사 및 연구하여 해당 기관의 시스템 보안 수준을 향상시키고, 취약점 진단 가이드에 따른 시스템 취약점 진단 도구 개발 및 접속 로그를 기반으로 하는 이상 접속 행위를 탐지하는 도구를 통해 간편한 점검을 돕기 위하여 주제로 선정하게 되었다.

2. 관련연구

2.1 Shell Script

Shell Script는 Unix와 Linux 운영 체제에서 명령어와 스크립트 문법을 사용하여 작성되는 스크립트 파일이다. 셸이나 명령 줄 인터프리터에서 돌아가도록 작성할 수 있고, 자동으로 유저 및 그룹, 권한을 설정하거나 코드를 실행하기 위한 라이브러리를 설치하는 등 다양한 문제를 해결하기 위해서 사용되며 간단한 데이터 분석을 위해서도 사용할 수 있다.

2.2 Python

Python 언어는 소프트웨어 개발, 데이터 분석, 기계 학습 등에 사용되는 프로그래밍 언어로, 여러 플랫폼에서 실행될 수 있고 거의 모든 유형의 시스템과 원활하게 호환되며 효율적이고 배우기 쉬운 언어이다. 또한 Python은 객체 지향 기반 인터프리터 방식의 언어로 결과를 바로 확인할 수 있다.

본 연구에서 활용한 Python의 모듈은 다음과 같다.

- Tkinter : Python에서 GUI를 개발할 때 사용되는 모듈로, Python에 기본적으로 탑재되어 있다.

- Paramiko : Python에서 SSH 클라이언트를 만들기 위한 라이브러리로, 안전하게 원격 서버에 연결하고 명령을 실행할 수 있게 해주는 도구이다. Paramiko를 통해 원격 서버나 네트워크 장비를 제어하거나 자동화 작업을 수행할 수 있다.

- Subprocess : Python에서 외부 프로세스를 실행하고 관리하는데 사용되는 모듈로 다른 프로그램을 호출하고 명령어를 실행하며 프로세스와 상호작용을 통해 Python의 기능을 확장할 수 있도록 한다. Python에서 시스템 레벨의 작업을 수행할 때 필수적으로 사용되며 다양한 운영체제에서도 호환된다.

- Shutil : Python으로 파일 또는 폴더를 이동시킬 때 사용하는 모듈이다.

2.3 Windows

Windows 시스템은 Microsoft에서 개발한 컴퓨터 GUI 운영체제로, 컴퓨터 역사상 가장 많이 사용되고 있는 운영체제이다. 일반 사용자들에게 매우 익숙할 뿐 아니라 호환되는 응용 프로그램이 많다는 장점이 있다.

2.4 Linux

Linux는 1991년 리누스 토르발스가 출시한 운영 체제 커널인 리눅스 커널에 기반을 둔 오픈 소스 유닉스 계열 운영 체제 계열이다. 리눅스는 일반적으로 리눅스 배포판 안에 패키지 처리된다.

2.5 Unix

Unix는 교육 및 연구 기관에서 즐겨 사용되는 범용 다중 사용자 방식의 대화식, 시분할 처리 시스템용 운영 체제이다. 1970년대 초반 처음 개발되었으며 여러 회사들과 비영리 단체들이 이 커널로 활용하여 다양한 운영체제를 개발하고 있다.

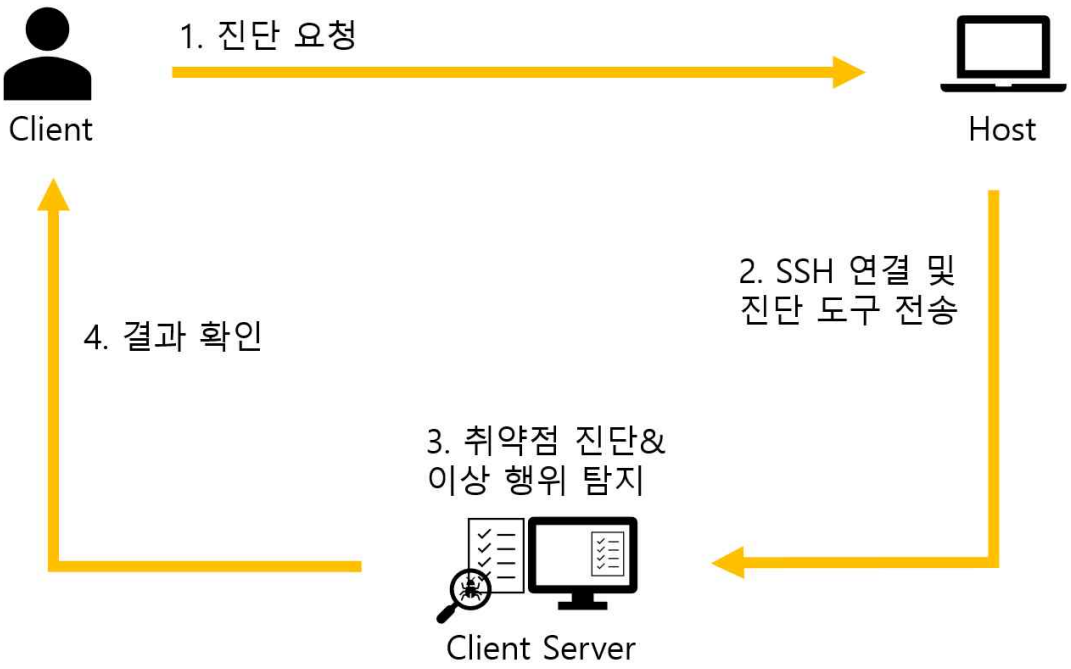
2.6 SSH

SSH 파일 전송 프로토콜은 신뢰할 수 있는 데이터스트림을 통해 파일 접근, 파일 전송, 파일 관리를 제공하는 네트워크 프로토콜이다. 국제 인터넷 표준화 기구(IETF)가 보안 파일 전송 기능을 제공할 목적으로 시큐어 셸 프로토콜(SSH) 버전 2.0의 확장으로 설계하였다. IETF 인터넷 초안에 따르면 이 프로토콜이 SSH-2 프로토콜의 문맥 안에 기술되어 있지만 전송 계층 보안(TLS)를 통하는 보안 파일 전송 프로그램이나 VPN 응용 프로그램의 관리 정보 전송과 같은 다른 수많은 응용 프로그램에도 사용할 수 있다고 언급되어 있다.

3. 본 론

3.1 시스템 구성

Client는 Host에 진단을 요청할 수 있고, 진단을 요청받은 Host가 SSH 연결 및 진단 도구를 전송한다. 이후 Client Server에서 시스템 취약점 진단 및 이상 접속 행위를 탐지하고 이후 결과를 보고서의 형태로 Client가 확인할 수 있다.



[그림 1. 전체적인 구상도]

3.2 프로그램 구성

프로그램은 크게 취약점 진단 스크립트, 이상 행위 탐지 스크립트, 진단 GUI로 구성되어 있다.

3.2.1 진단 스크립트

서버 취약점 진단 자동화 프로그램은 Liunx 서버를 자동으로 진단하기 위한 진단 스크립트로, 주요정보통신기반시설 기술적 취약점 분석평가 방법 상세가이드를 참고하여 개발했으며, 진단 항목은 다음과 같다.

- 계정 관리 : 불필요한 계정 제거 외 14개 항목

```
#echo "항목 코드 U-01"

SECURETTY=/etc/securetty
SECURE=$(cat $SECURETTY | grep -i "pts" | wc -l)
SSH=/etc/ssh/sshd_config


if [ -e $SECURETTY ] && [ -e $SSH ]; then
        SSHD=$(cat $SSH | grep -E "PermitRootLogin yes|PermitRootLogin no" | awk '{print $2}')
        if [ $SECURE != 0 ] && [[ $SSHD == "yes" ]] ; then
                echo "U-01 취약 상" >> result.csv
        elif [ $SECURE -eq 0 ] && [[ $SSHD == "yes" ]] ; then
                echo "U-01 취약 상" >> result.csv
        else
                echo "U-01 양호 상" >> result.csv
        fi
else
        echo "U-01 점검 상" >> result.csv
fi

#echo "항목 코드 U-02"

PW=/etc/security
PWD=/etc/security/pwquality.conf
MIN=$(cat $PWD | grep "minlen" | awk '{print $4}')

if [ -e $PW ]; then
        if [[ ${MIN} -ge 8 ]]; then
                echo "U-02 양호 상" >> result.csv
        else
                echo "U-02 취약 상" >> result.csv
        fi
else
        echo "U-02 점검 상" >> result.csv
fi

#echo "항목 코드 U-03"
```

[그림 2. 계정관리 항목 취약점 진단 코드]

- 파일 및 디렉터리 관리 : UMASK 설정 관리 외 18개 항목

```
#echo "항목 코드 U-05"

PH=$(echo $PATH | egrep "\.:|::|:.:")

if [ -z $PH ]; then
        echo "U-05 양호 상" >> result.csv
else
        echo "U-05 취약 상" >> result.csv
fi

#echo "항목 코드 U-06"

DIR=$(find / \( -nouser -o -nogroup \) -print 2>/dev/null)

if [ -z $DIR ]; then
        echo "U-06 양호 상" >> result.csv
else
        echo "U-06 취약 상" >> result.csv
fi

#echo "항목 코드 U-07"

PWD=$(ls -l /etc/passwd | awk '{print $3}')
PSWD=$(stat -c "%a" /etc/passwd)

if [ "$PWD" == "root" ]; then
        if [ $PSWD -le 644 ]; then
                echo "U-07 양호 상" >> result.csv
        else
                echo "U-07 취약 상" >> result.csv
        fi
else
        echo "U-07 취약 상" >> result.csv
fi
```

[그림 3. 파일 및 디렉터리 관리 취약점 진단 코드]

- 서비스 관리 : FTP 서비스 확인 외 34개 항목

```
#!/bin/bash

#echo "항목코드 U-19"

ls -lL /etc/xinetd.d |grep finger > /dev/null 2>&1

if [ $? -eq 0 ]; then
        echo "U-19취약 상" >> result.csv
else
        echo "U-19 양호 상" >> result.csv
fi

#echo "항목코드 U-20"

if [ -f /etc/vsftpd/vsftpd.conf ]; then
        if grep -q "ftp" /etc/vsftpd/vsftpd.conf; then
                echo "U-20 양호 상" >> result.csv
        else
                echo "U-20 취약 상" >> result.csv
        fi
else
        echo "U-20 점검 상" >> result.csv
fi

#echo "항목코드 U-21"

files=$(ls -alL /etc/xinetd.d/* /dev/null 2>&1 | egrep "rsh|rlogin|rexec" | egrep -v "grep|klogin|kshell|kexec")

if [ -n "$files" ]; then
        echo "U-21 취약 상" >> result.csv
else
        echo "U-21 양호 상" >> result.csv
fi

#echo "항목코드 U-22"

cron_file="/etc/crontab"
cron_owner=$(stat -c %U "$cron_file")
cron_permissions=$(stat -c %a "$cron_file")
```

[그림 4. 서비스 관리 취약점 진단 코드]

- 패치 관리 : 최신 보안패치 및 벤더 권고사항 적용

- 로그 관리 : 로그의 정기적 검토 및 보고 외 1개 항목

```
#!/bin/bash

#echo "항목 코드 U-42"
echo "U-42 취약 상 ">> result.csv

#echo "항목 코드 U-43"
echo "U-43 점검 상" >> result.csv

#echo "항목 코드 U-72"
echo "U-72 점검 하" >> result.csv
```

[그림 5. 패치 관리, 로그 관리 취약점 진단 코드]

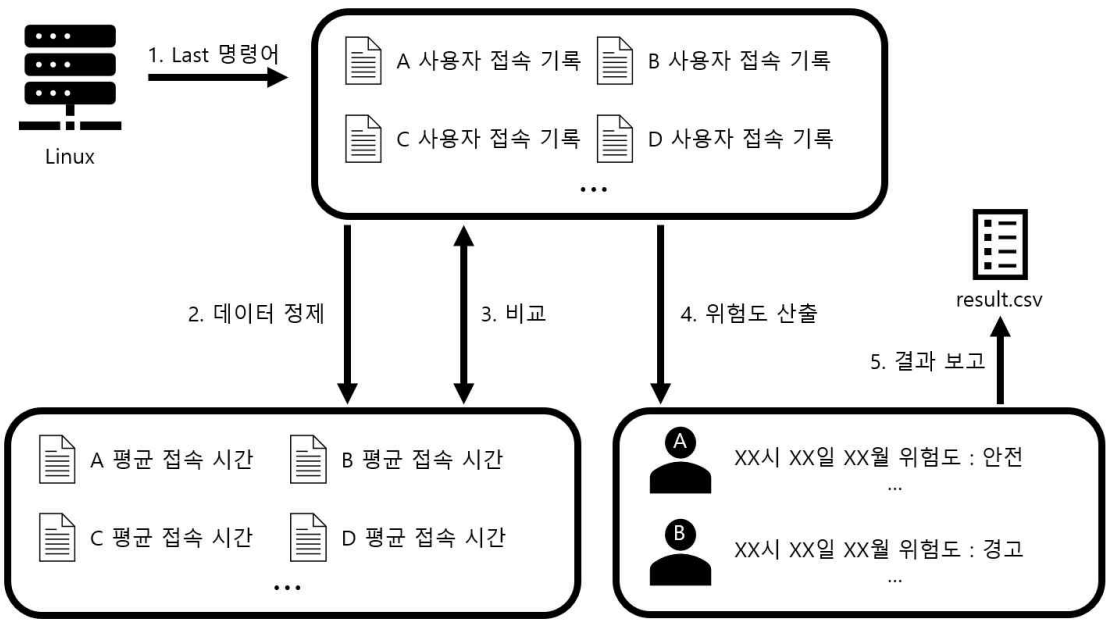
- 취약할 경우 취약점에 대한 조치 방법 제시

```
#!/bin/bash

if grep -qF "01" error.txt; then
        echo "U-01 조치 방법 안내" >> finish.csv
        echo "root 계정 원격접속 제한" >> finish.csv
        echo -e "[Telnet 서비스 사용시]\n 1. '/etc/securetty' 파일에서 pts/x 설정 제거 또는, 주석 처리\n 2. '/etc/pam.d/login' 파일 수정 또는, 신규 삽입\n (수정 전) #auth required /lib/secu
rity/pam_securetty.so\n (수정 후) auth required /lib/security/pam_securetty.so" >> finish.csv
        echo -e "[SSH 서비스 사용시]\n 1. vi편집기를 이용하여 '/etc/ssh/sshd_config' 파일 열기\n 2. 아래와 같이 주석 제거 또는, 신규 삽입\n (수정 전) #PermitRootLogin Yes\n (수정 후) Permit
RootLogin No" >> finish.csv
        echo "" >> finish.csv
fi

if grep -qF "02" error.txt; then
        echo "U-02 조치 방법 안내" >> finish.csv
        echo "패스워드 복잡성 설정" >> finish.csv
        echo -e "1. 패스워드 복잡성 설정 파일 확인\n #/etc/security/pwquality.conf 파일 수정\n 2. 패스워드 정책을 설정함\n password requisite pam_cracklib.so try_first_pass retry=3 minlen=8
 lcredit=-1 ucredit=-1 dcredit=-1 ocredit=-1" >> finish.csv
        echo "" >> finish.csv
fi

if grep -qF "03" error.txt; then
        echo "U-03 조치 방법 안내" >> finish.csv
        echo "계정 잠금 임계값 설정" >> finish.csv
        echo -e "1. vi 편집기를 이용하여 '/etc/pam.d/system-auth' 파일 열기\n 2. 아래와 같이 수정 또는, 신규 삽입\n auth reqired /lib/security/pam_tally.so deny=5 unlock_time=120 no_magic_r
oot\n account required /lib/security/pam_tally.so no_magic_root reset" >> finish.csv
        echo "" >> finish.csv
fi

if grep -qF "04" error.txt; then
        echo " U-04 조치 방법 안내" >> finish.csv
        echo "패스워드 파일 보호" >> finish.csv
        echo -e "1. #pwconv ---> 쉐도우 패스워드 정책 적용 방법\n 2. #pwunconv ---> 일반 패스워드 정책 적용 방법" >> finish.csv
        echo "" >> finish.csv
fi

if grep -qF "05" error.txt; then
        echo "U-05 조치 방법 안내" >> finish.csv
        echo "root홈, 패스 디렉터리 권한 및 패스 설정" >> finish.csv
        echo -e "1. vi 편집기를 이용하여 root 계정의 설정 파일(~/.profile 과 /etc/profile) 열기\n 2. 아래와 같이 수정\n (수정 전) PATH=.:$PATH:$HOME/bin\n (수정 후) PATH=$PATH:$HOME/bin:."
>> finish.csv
```

[그림 6. 취약할 경우 취약점에 대한 조치 방법 제시에 관한 코드]

3.2.2 이상행위탐지

Linux 시스템에서 사용자 접속을 기록하는 Last 명령어를 사용하여 각 사용자의 접속 기록을 문서화하고 사용자의 평균 접속 시간을 계산하여 접속 기록과 비교 후 평균 접속 시간대와 크게 벗어난 기록에 따라 '안전', '경고', '위험'의 3단계로 산출하여 보고서의 형태로 문서화할 수 있도록 제작하였다.



[그림 7. 이상행위탐지 구상도]

```
users=$(grep /bin/bash /etc/passwd | cut -f1 -d:)
echo "사용자,로그인 날짜(월),로그인 날짜(일),로그인 시각(시),위험도" >> result.csv
sed -i '1s/^/\n/' result.csv
rm userstime.txt
for user in $users; do
  last "$user" | awk '{if($1=="'$user'") {print $4, $5, $6, $7, $8, $9}}' | awk '!/crash/ && !/still/ && !/logged/ {print}' > "${user}.txt"
  if [ -s "${user}.txt" ]; then
    echo "${user}.txt save to file completely"
  else
    echo "${user}.txt save to file False"
  fi
  file="${user}.txt"
  login_times=($(awk '{split($4, a, ":"); print a[1]}' "$file"))
  login_months=($(awk '{print $2}' "$file"))
  login_dates=($(awk '{print $3}' "$file"))
  logout_times=($(awk '{split($6, a, ":"); print a[1]}' "$file"))
  total_login=0
  total_logout=0
  for time in "${login_times[@]}"; do
    ((total_login+=time))
  done
  for time in "${logout_times[@]}"; do
    ((total_logout+=time))
  done
  if [ "${#login_times[@]}" -gt 0 ]; then
    avg_login=$((total_login / ${#login_times[@]}))
  else
    avg_login=0
  fi
  if [ "${#logout_times[@]}" -gt 0 ]; then
    avg_logout=$((total_logout / ${#logout_times[@]}))
  else
    avg_logout=0
  fi
  echo "사용자: $user" >> userstime.txt
  echo "로그인 평균 시간: $avg_login" >> userstime.txt
  echo "로그아웃 평균 시간: $avg_logout" >> userstime.txt
  for ((i=0; i<${#login_times[@]}; i++)); do
```

[그림 8. 이상행위탐지 코드]

3.2.3 진단 GUI

프로그램을 실행하면 좌측부터 해당 프로그램에 대한 간략한 소개와 본격적인 시스템 취약점 진단을 위한 SSH 연결, 스크립트 전송 및 실행, 결과 확인을 할 수 있도록 단추와 직관적인 설명이 기재되어 있고 본 연구를 기획, 개발한 제작자의 설명도 포함시켜 제작하였다.

[그림 9. 프로그램 GUI의 모습]

```
#!/usr/bin/python3

import tkinter as tk
from tkinter import *
from tkinter import ttk
import paramiko
import subprocess
import os
import tkinter.messagebox as messagebox
import shutil

import requests
from io import BytesIO
from PIL import Image, ImageTk
import tkinter.font

# MOGLOG v1.0 GUI 창 생성
window = tk.Tk()
window.title("MOGLOG v1.0")
window.geometry("1100x650")  # 창 크기 설정

# SSH 연결 정보 (미리 초기화)
ip_address = "192.168.129.140"
username = "root"
password = "cutyyoon7235!"
ssh_client = None
result_output = ""

# 배경 캔버스 생성
bg_canvas = tk.Canvas(window, bg="white",width=1800, height=600)
bg_canvas.pack()

# "소개" 프레임 생성
intro_frame = tk.Frame(bg_canvas)
intro_frame.pack(side="left",padx=8, pady=10)  # 왼쪽 상단에 붙이기

# "소개" 레이블 생성
intro_label = tk.Label(intro_frame, text="소개", bg="gray", width=50, height=2)
intro_label.config(font=("Helvetica", 10, "bold"))
intro_label.pack()
```

[그림 10. GUI 코드 일부]

4. 분 석

4.1 활용 결과 및 성능

프로그램 활용 결과 해당 시스템에 대한 총 72여가지의 취약점 진단이 가능하고 취약한 경우에는 보완 방법도 제시할 수 있다. 또한 각 사용자들의 평균 접속 시간에 대한 이상 접속 행위로 의심되는 활동을 분석할 수 있다.

4.2 추후 보완사항

이후 지속적인 업데이트를 통하여 새롭게 발견되는 취약점에 대한 진단 항목을 추가할 수 있고, Linux 시스템 뿐만이 아닌 Windows 시스템의 취약점 진단 도구도 포함할 수 있

다. 또한 이상 행위에 대한 기준을 관리자가 보다 세밀하게 설정하여 분 단위나 '경고', '위험'의 기준에 대한 값을 변경할 수 있도록 보완할 예정이다.

5. 결 론

5.1 결론

이번 Linux 시스템의 취약점 진단 도구 개발 및 이상 행위 탐지 도구 개발 연구를 통해 Linux 운영체제와 각 진단 항목에 대한 이해도를 높였으며 Linux 시스템의 명령어 및 스크립트 개발에 대한 숙련도가 증가하였고, 계정 관리 및 권한 설정의 중요성을 깊게 탐구하는 계기가 되었다. 또한 취약점 진단을 통한 공격 피해를 예방할 수 있게 되었다.

5.2 기대효과

현대 사회에서 보안 문제는 점점 더 중요해지고 있다. 사이버 공격, 데이터 유출, 악성 코드 등으로부터의 보호가 필요한 기업 및 개인이 늘어나고 이에 따라 시스템 취약점 진단과 접속 로그 기반 이상행위 탐지 솔루션은 신속한 보안 대응이 가능할 것으로 보인다.

6. 별 첨

6.1 팀원 소개

강성현 (중부대학교 정보보호학전공) : 이상 행위 탐지 도구 개발
김고운 (중부대학교 정보보호학전공) : 취약점 진단 도구 개발
송소연 (중부대학교 정보보호학전공) : 이상 행위 탐지 도구 개발
윤지예 (중부대학교 정보보호학전공) : 취약점 진단 도구 개발
정지수 (중부대학교 정보보호학전공) : 취약점 진단 도구 개발

6.2 소스 코드

(https://github.com/shg4679/Linux-vulnerability-Diagnosis)