Server Sentry Tool을 탑재한 IDS 구축

팀 명 : S.S
지도 교수 : 양환석 교수님
팀 장 : 김의선
팀 원 : 안재민
황규남
손솔미
김미리

2016. 05
중부대학교 정보보호학과

목 차

1. 서론
    1.1 IDS 란?
    1.2 IDS 의 기능과 분류
    1.3 주제 선정

2. 관련연구
    2.1 쉘 프로그래밍
    2.2 QT 프로그래밍 (C++)
    2.3 웹 프로그래밍
    2.4 Kail Linux
    2.5 R 언어

3. 본론
    3.1 프로그램의 구성
    3.2 세부 구조
    3.3 QT를 이용한 GUI 환경 구축
    3.4 소스 코드

4. 결론

5. 참고 자료

6. 발표 자료

1. 서론

1.1 IDS 란?

IDS는 Intrusion Detection System 의 약자로 침입탐지시스템을 의미한다. IDS는 리눅스 방화벽이나 내부 보호망 보호를 수행하는데 있어서 발생하는 한계를 보안하고 네트워크 및 시스템의 사용을 실시간으로 모니터링하고 침입을 탐지하는 시스템을 말한다.
IDS 는 단순한 시스템 제어 기능 뿐 만 아니라 침입 패턴을 이용한 보안 시스템과 허가 되지 않은 사용자에 대한 접속 제한, 주요 시스템 파일의 위•변조, 시스템에 피해를 주는 행위를 감시하여 초기에 대응하고 실시간 처리를 목적으로 두고 있다.

1.2 IDS 의 기능과 분류

IDS 의 가장 일반적인 기능은 시스템의 변경을 탐지하는 것이다.
리눅스 기존 방화벽이 탐지 할 수 없는 악의적 접속 및 공격과 포트 및 프로토콜을 이용한 네트워크 공격에 대해 감시와 사용자 알람을 지원한다. IDS 는 침입 탐지 기법에 따라 나누어지는데 통계적 자료를 근거로 한 비정상적인 탐지 기법과 해킹 공격 방법을 기반으로 침입 패턴에 따른 분석 결과를 이용하여 탐지하는 오용 탐지 기법이 있다.
대부분의 IDS 는 침입 패턴에 따른 오용 탐지 기법을 선택하여 상용화하고 있지만 조금이라도 변형된 공격에 대해 탐지하지 못한다는 치명적인 단점을 지니고 있다.

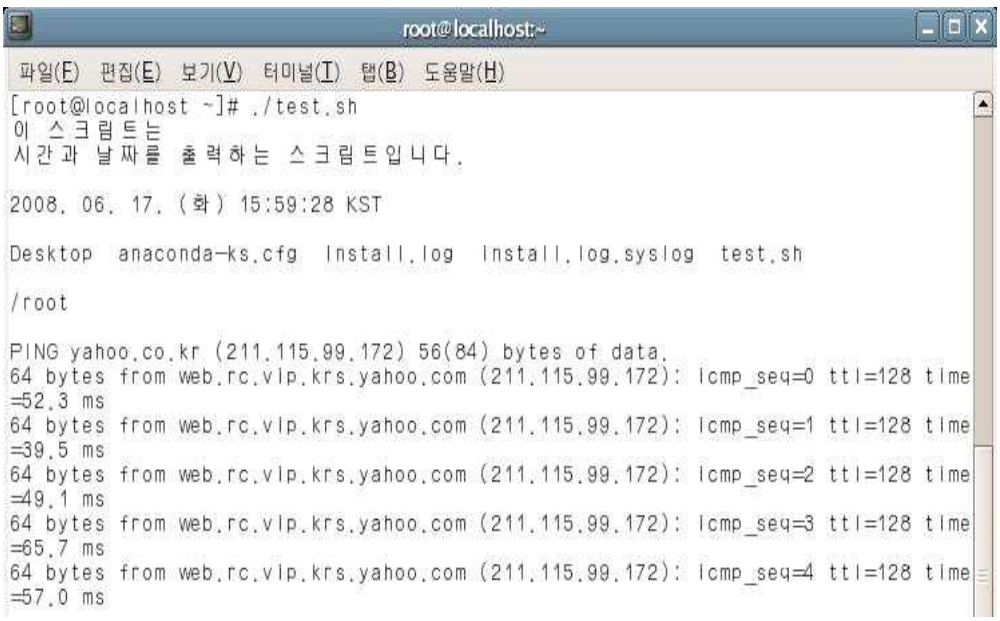
1.3 주제 선정 이유

IDS 는 매우 다양하고 많이 쓰이고 있다. 하지만 기존의 IDS 는 많이 알려진 만큼 기존 보안 패턴에서 벗어난 공격에 대해 취약 하다는 문제점을 지니고 있다. 모의 해킹을 통해 이러한 취약점의 보안 방법을 연구해보았으며 기존 터미널 위주의 사용법이 복잡한 보안 시스템의 틀에서 벗어나 사용이 편리한 그래픽 기반 인터페이스 방식 IDS를 제작해보기 위해 이러한 주제를 선정하였다.

2. 관련연구

2.1 쉘 프로그래밍

쉘이란 이용자와 시스템 간의 대화를 가능하게 해주며 이용자가 입력한 문장을 읽어 그문장이 요청하는 시스템 기능을 수행하도록 해주는 명령어 해석기이다.
쉘의 종류로는 Bourne shell, C shell Korn shell Bash 쉘이 있고 fedora 11에서는 bash shell을 사용 한다. 쉘 스크립트는 리눅스 시스템에서 지원하는 명령어들을 집합으로 묶어 프로그램화 한 것이고 쉘 스크립트에서 사용한 문법은 조건문, 반복문(if,while) 등을 사용하였다.



< 그림 2 – 1 : 쉘 스크립트 >

2.2 QT 프로그래밍

Qt는 소스코드 하나로 윈도우, MAC OS ,Linux 와 같은 다양한 운영체제에서 모두 실행되는 GUI 응용프로그램을 제작 할 수 있는 C++ 응용프로그램 개발 프레임 워크이다. 서버용 콘 솔과 명령 줄 도구등

과 같은 곳에 사용되고 아래 사진과 같이 개발자가 원하는 대로 사용할수 있다. 프레임 워크란 소프트웨어 어플리케이션이 개발을 수월하게 하기 위해 소프트웨어의 구체적 기능들에 해당하는 부분의 설계와 구현을 재사용 가능하도록 협업화된 형태로 제공하는 소프트웨어 환경이다.
기존의 QT는 C++ 소스코드만으로 개발하였으나 비주얼 디자인 기능을 가진 QT Disigner 프로그램을 사용하여 좀더 편리하게 GUI를 제작할수 있다.

< 그림 2 – 2 : QT 프로그래밍으로 제작한 IDS 툴 >

2.3 웹 프로그래밍

웹 상에서 사용자와 기업 또는 사용자들 간의 연결을 가능하게 하는 웹 기반의 프로그래밍 언어로, HTML, XML, JavaScript 언어를 사용하였다. HTML은 웹에서의 정보를 표현하고 홈페이지를 만들기 위해 사용되는 언어로, SGML에서 하이퍼텍스트를 사용하는 문서를 작성할 수 있도록 만들어진 언어이며, 태그라는 명령어를 사용해 다양하게 표현할 수 있다. 또한 링크라는 기능을 사용하여 홈페이지 안의 특정 장소

를 클릭하면 관련된 다른 페이지나 다른 홈페이지로 간단하게 이동할 수 있다.

JavaScript는 HTML 문서의 정적인 한계를 극복하기 위해 만든 것으로 브라우저 자체에 내장된 해석기능을 이용한 클라이언트 기반의 일종의 스크립트 언어이다.

자바의 언어 규격으로부터 정수형이나 문자열형 등의 변수의 형을 생략하거나 새로운 클래스 정의를 할 수 없도록 하였다. 스크립트는 HTML 문서 속에 직접 기술하며, 'script'라는 꼬리표를 사용하고, 작고 빠르기 때문에 웹문서를 동적으로 꾸밀 때 쓰인다.

XML은 인터넷 웹페이지를 만드는 HTML을 획기적으로 개선하여 만든 언어이다. 홈페이지 구축기능, 검색기능 등이 향상되었고, 클라이언트 시스템의 복잡한 데이터 처리를 쉽게 하며, HTML은 태그의 종류가 한정되어 있는 반면 XML은 문서의 내용에 관련된 태그를 사용자가 직접 정의할 수 있으며, 그 태그를 다른 사람들이 사용하도록 할 수 있어 인터넷 사용자가 웹에 추가할 내용을 작성, 관리하기 쉽게 되어 있다.

< 그림 2 – 3 : 웹 프로그래밍으로 제작한 홈페이지 화면 >

2.4 칼리리눅스

칼리리눅스는 Offensive Security에서 만든 Backtrack의 후속버전으로 각종 해킹툴을 모아놓은 운영체제이다 BackTrack과 비슷하지만 칼리리눅스의 특징은 BackTrack에서 동작하지 않는것과 중복되는것을 제거후 수백개의 침투테스트 툴이 있으며 BackTrack 에서 미약했던 무선장치 지원을 강화하였다 .
또한 BackTrack 은 우분투를 기반으로 만들어 졌고 칼리리눅스는 데비안을 기반으로 만들어졌다.
데비안 과 우분투는 리눅스 종류 로 데비안은 데비안은 자유 소프트웨어를 개발하고 자유 소프트웨어 커뮤니티의 이상을 널리 알리는 일을 위해 조직한 자원자로만 구성된 조직이 참여한 데비안 프로젝트에서 만들어 배포하는 공개 운영 체제이고.
우분투는 데비안 GNU/리눅스(Debian GNU/Linux)에 기초한 컴퓨터 운영 체제로서 고유한 데스크탑 환경인 유니티를 사용하는 리눅스 배포판이다. GNU리눅스는
칼리 리눅스는 설치하지 않아도 부팅 가능한 칼리 리눅스 이미지만 있으면 사용할수 있다.

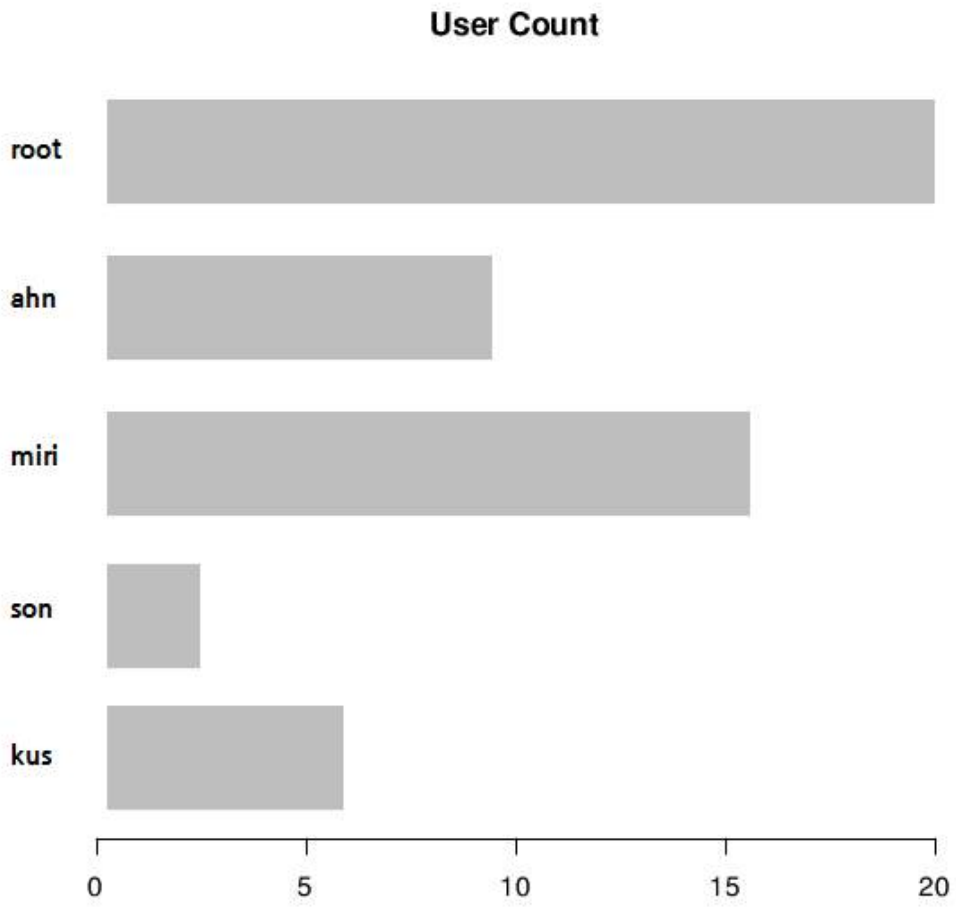
< 그림 2 – 4 : 칼리 리눅스 >

2.5 R언어

R언어란 개발된 통계 및 그래프 작업을 위한 인터프리터 프로그래밍 언어이다

R은 다양한 통계 및 그래픽 기술을 제공하고, 확장성이 뛰어나고 R은 그오픈 소스 경로를 제공 R의 장점 중 하나는 손쉽게 필요한 곳에 수학기호와 수식을 포함한, 잘 디자인된 출판물 수준의 plot을 생산합니다.

R은 다양한 종류의 UNIX 플랫폼과 유사 시스템(FreeBSD와 Linux 등), 윈도우와 맥 OS에서 컴파일되고 실행할 수 있습니다.



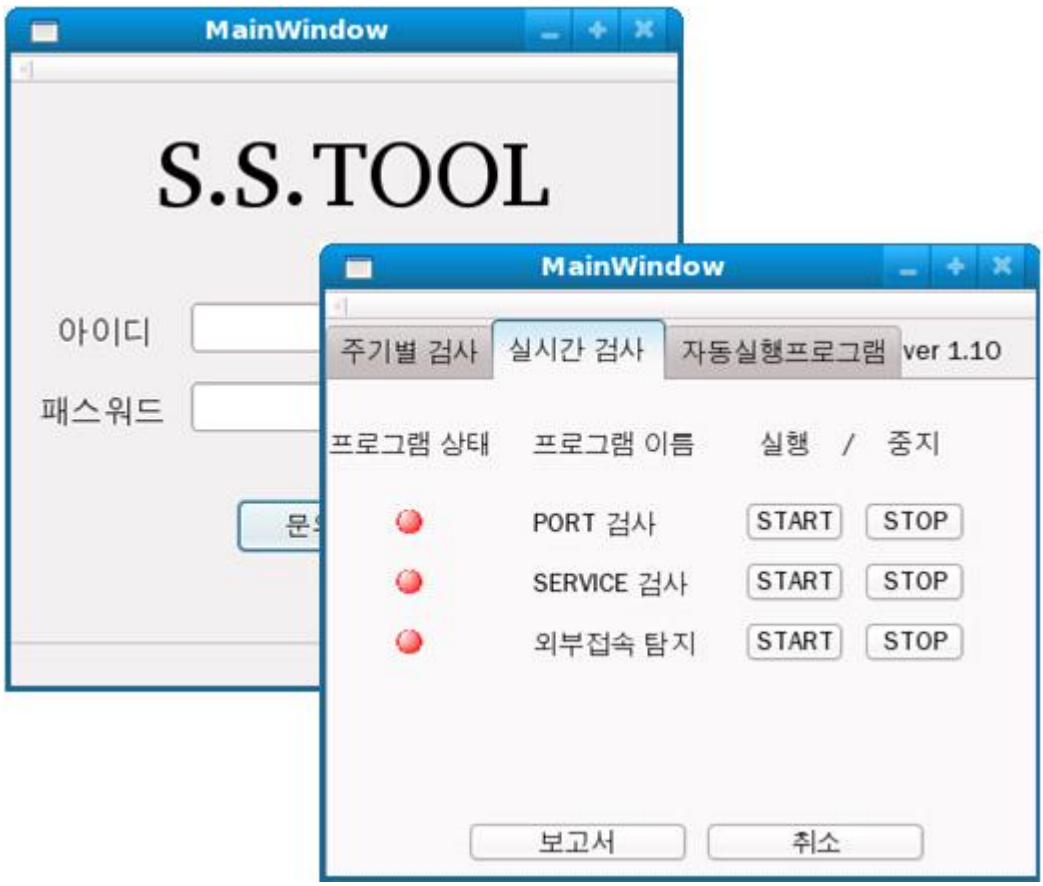
< 2 – 5 : R 언어를 이용한 로그인 관리 그래프 >

3. 본론

3.1 프로그램의 구성

- 메인 관리자 PC 와 그래프 모니터링 PC, Web서버 PC, 모의해킹용 Kail Linux PC 로 구성되어있다.
- 프로그램은 크게 주기별, 실시간, 자동 실행 3가지로 나누어진다.
- 시스템 탐지에 대한 결과는 관리자에게 알림으로 알려준다.
- 로그인 인증을 사용하여 정해진 관리자만이 사용할 수가 있다.

관리자는 Root 계정의 ID와 PW를 통하여 프로그램을 실행하게 된다. 프로그램 실행 후에는 기존의 터미널 작업이 아닌 GUI 환경에서 PC를 모니터링 하며 관리 할 수 있다.

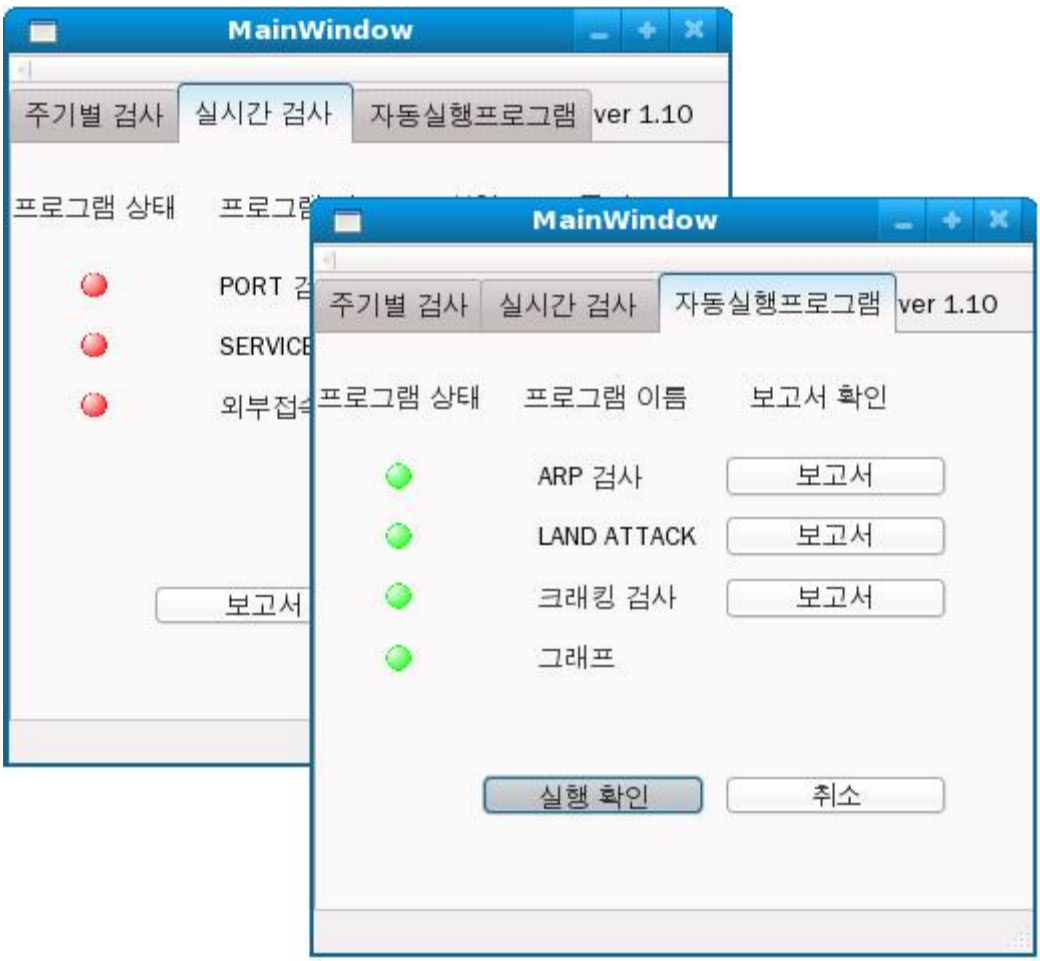


< 그림 3 – 1 : 로그인을 통한 관리자 인증 >

3.2 세부 구조

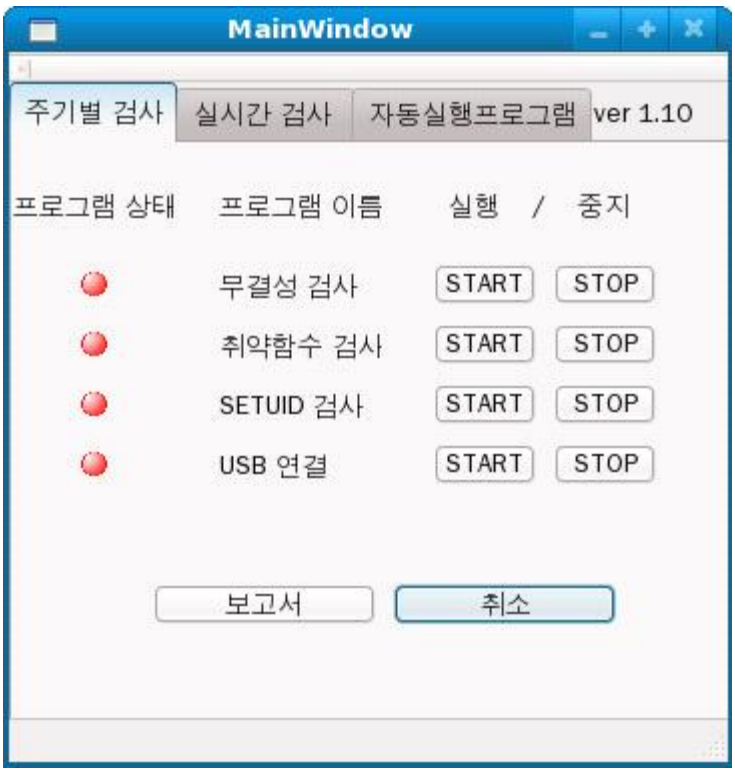
프로그램은 실시간 탐지, 주기별 탐지, 자동 실행 3가지로 나눌 수 있다. 탐지 기준을 나누는 가장 큰 차이점은 탐지 기법으로 나눌 수 있다.

자동실행 프로그램과 실시간 탐지는 침입 패턴에 따른 오용 탐지 기법으로 PC에 악영향을 주는 해킹 공격 탐지와 트래픽 사용량을 검사하여 문제가 발생 시 관리자에게 즉각적으로 알림 팝업을 띄우게 된다.
ARP Spoofing 공격, Land Attack 공격, Port 및 Service 검사, 크래킹 검사, 외부접속 탐지, 트래픽 그래프 등 이 있다.



< 그림 3 – 2 : 실시간 검사와 자동 실행 프로그램 목록 >

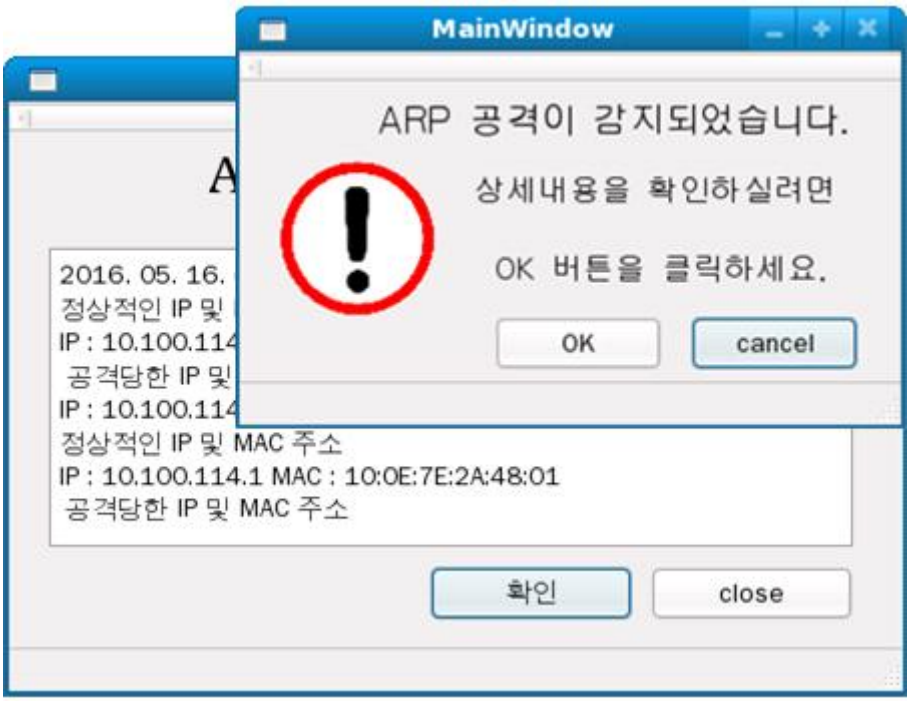
주기별 탐지는 시스템에 저장된 log 기록을 분석하여 사용자의 차단 및 관리하는 비정상적인 침입 탐지 기법으로 파일의 위•변조 검사 결과 및 해킹의 사전공격으로 예상되는 부분을 관리자에게 log 기록과 함께 알려주게 된다.
무결성 검사, 취약함수 검사, SETUID 검사, USB 연결 이 있다.



< 그림 3 – 3 : 주기별 검사 목로 >

3.2.1 ARP Spoofing 공격

ARP Spoofing 공격은 피해자의 MAC 주소를 갈취하여 공격하기 때문에 MAC 주소가 변경되는 것을 통하여 알 수 있다.
서버 라우터의 IP / MAC 주소 값을 미리 저장해둔 뒤비교하여 MAC 주소가 변경된 것을 확인 하였을 때 ARP 공격 알람을 띄운다.



< 그림 3 – 4 : ARP Spoofing 공격 감지 알람 과 보고서>

3.2.2 Land Attack 공격

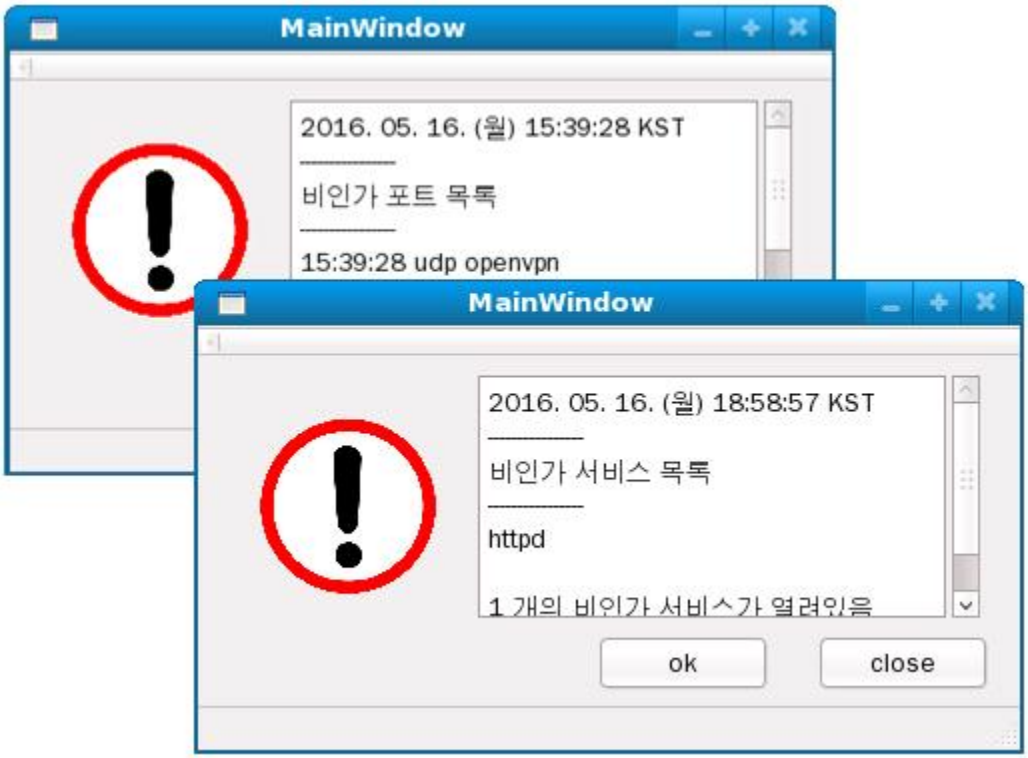
Land Attack 는 패킷이 전송 될 때 수신지와 송신지의 IP 값을 똑같이 만들어 공격 대상에게 보내게 된다. 이때 IP의 값은 피해자 자기 자신의 값으로 네트워크에 부하를 주는 DoS 공격이기 때문에 송신지와 수신지의 IP 값이 모두 자신의 IP 인 패킷을 확인하여 발견시 Land Attack 공격 알람을 관리자에게 띄운다.

< 그림 3 – 5 : Land Attack 감지 알림>

3.2.3 Port 및 Service 검사

Port와 Service 의 경우 직접 PC에 영향을 주는 것은 아니지만 허용되지 않은 임의 Port 및 Service 가 열려 있을 때는 해킹에 매우 취약한 상태가 되게 된다.

Port와 Service 의 상태를 실시간으로 확인하며 정해진 목록 이외의 임의의 조작이 있을 경우 관리자에게 즉각적으로 알림을 보내준다.

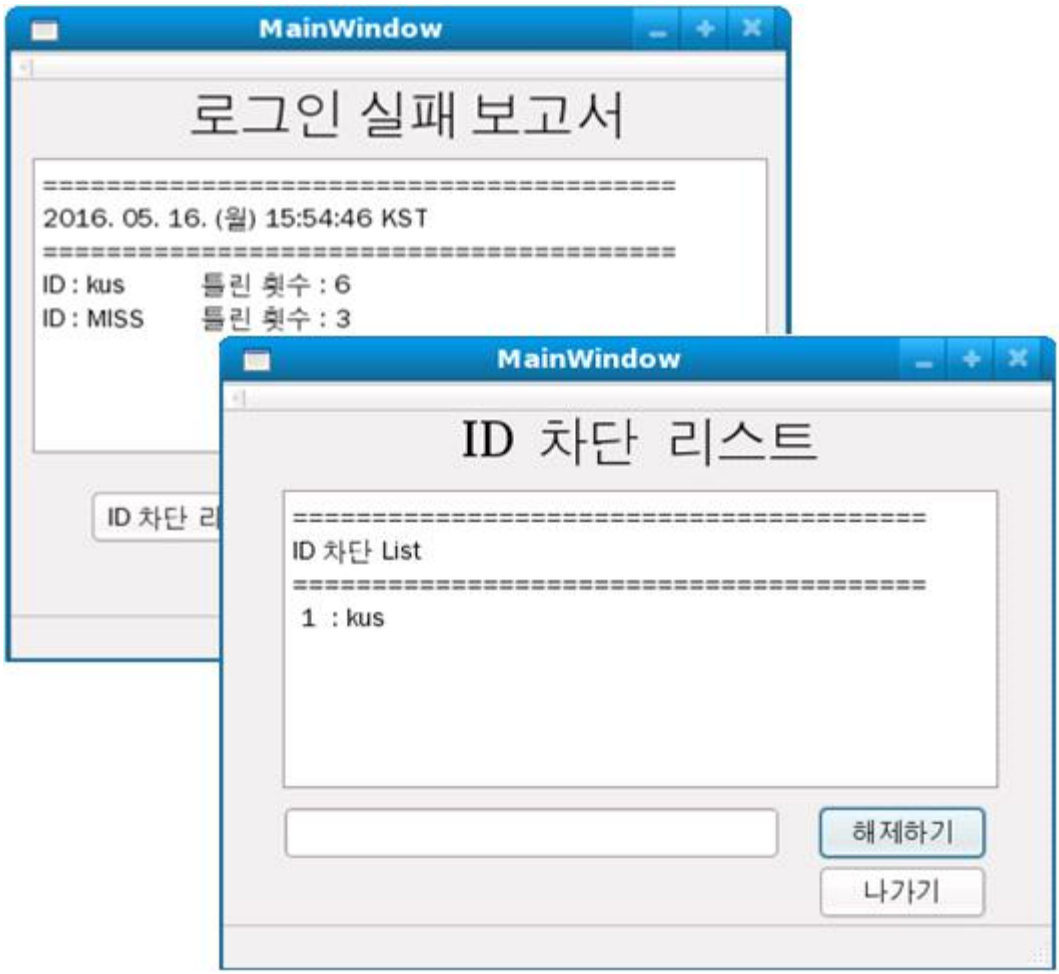


< 그림 3 – 6 : Port 및 Service 검사 알림 >

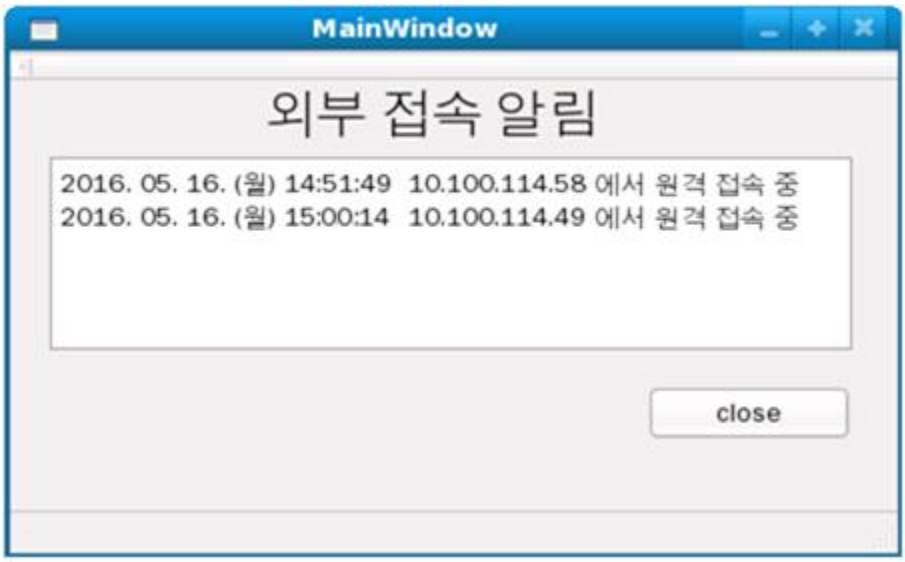
3.2.4 크래킹 검사 및 외부 접속 알림

리눅스는 기본적으로 원격 로그인에 대한 기록을 /var/log/secure 에 저장하게 되어 있다. 위 저장장소에 쌓인 log 기록을 확인하여 접속에 실패한 사용자를 분석하고 짧은 시간 내에 로그인 실패가 잦은 사용자의 IP 및 ID 차단을 한다. 또한 R 언어를 이용하여 그래프로 시각화 하여 관리자가 크래킹으로 의심 가는 계정에 대한 집중적 관리를 보다 쉽게 할 수 있도록 해준다.

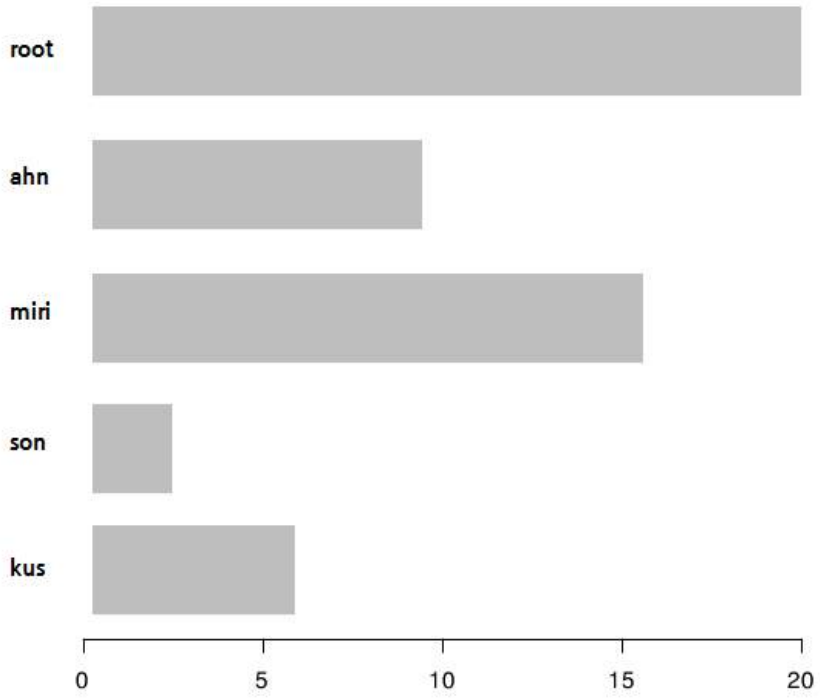
또한 외부 접속 알림은 사용자가 정상적으로 원격접속으로 접속하더라도 관리자에게 실시간 접속 상황을 알려주게 된다.



< 그림 3 – 7 : 로그인 실패 log와 ID, IP 차단 리스트 >



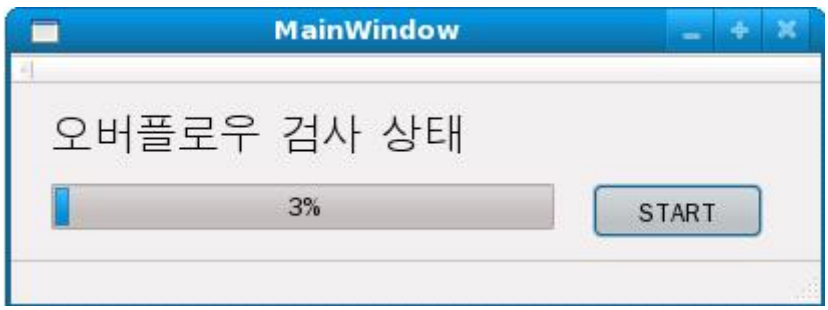
< 그림 3 – 8 : 외부 접속 알람 >



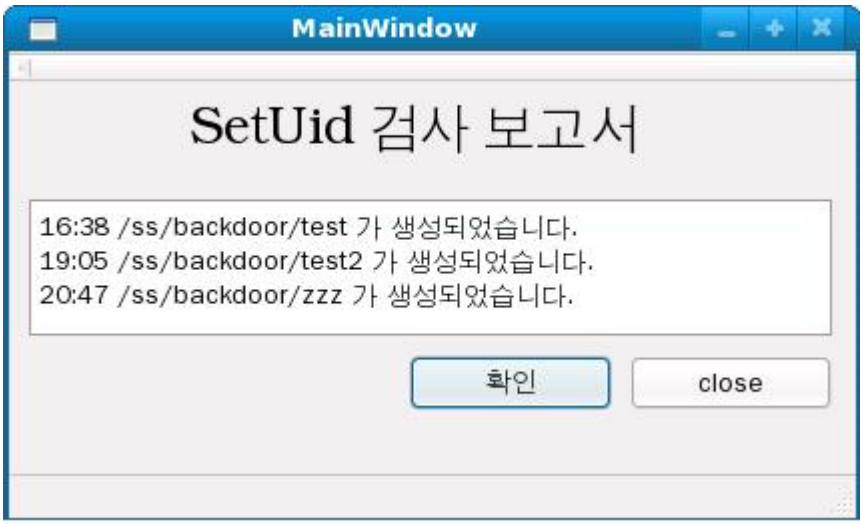
< 그림 3 – 9 : R 언어를 이용한 로그인 관리 그래프 >

3.2.5 무결성 & 취약함수 & SETUID

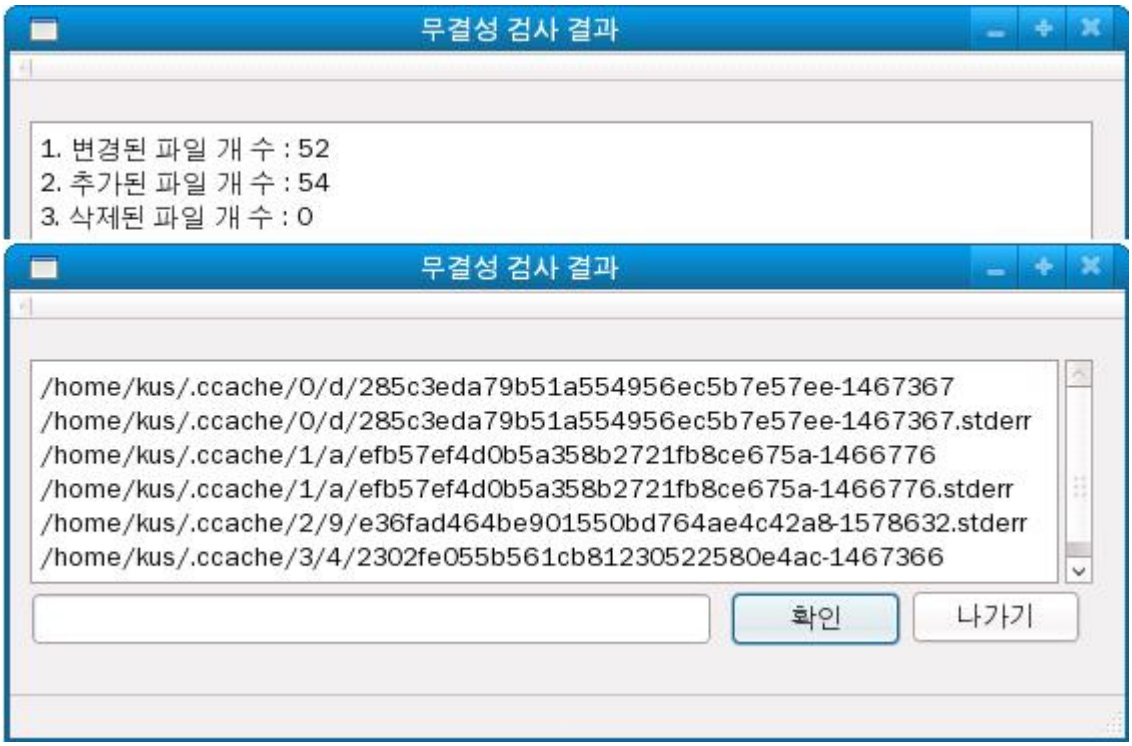
무결성 검사는 Fcheck를 이용하여 파일의 변동 내용을 확인한다. 무결성 검사는 단순한 파일의 변동 내용을 확인하는 것으로 그치지 않고 기존에 시간이 오래 걸리는 취약함수 검사나 SETUID 검사를 할 때 변경 내용이 있는 파일만 검사하여 보다 빠르게 확인 할 수 있게 한다. 취약함수 검사는 오버플로우 공격에 취약한 13개 함수를 선정하여 해당 함수를 포함한 파일이 존재하는지를 확인하게 된다.



< 3 – 10 : 취약함수(OverFlow) 검사 진행도>



< 그림 3 – 10 : SETUID 검사 보고서>

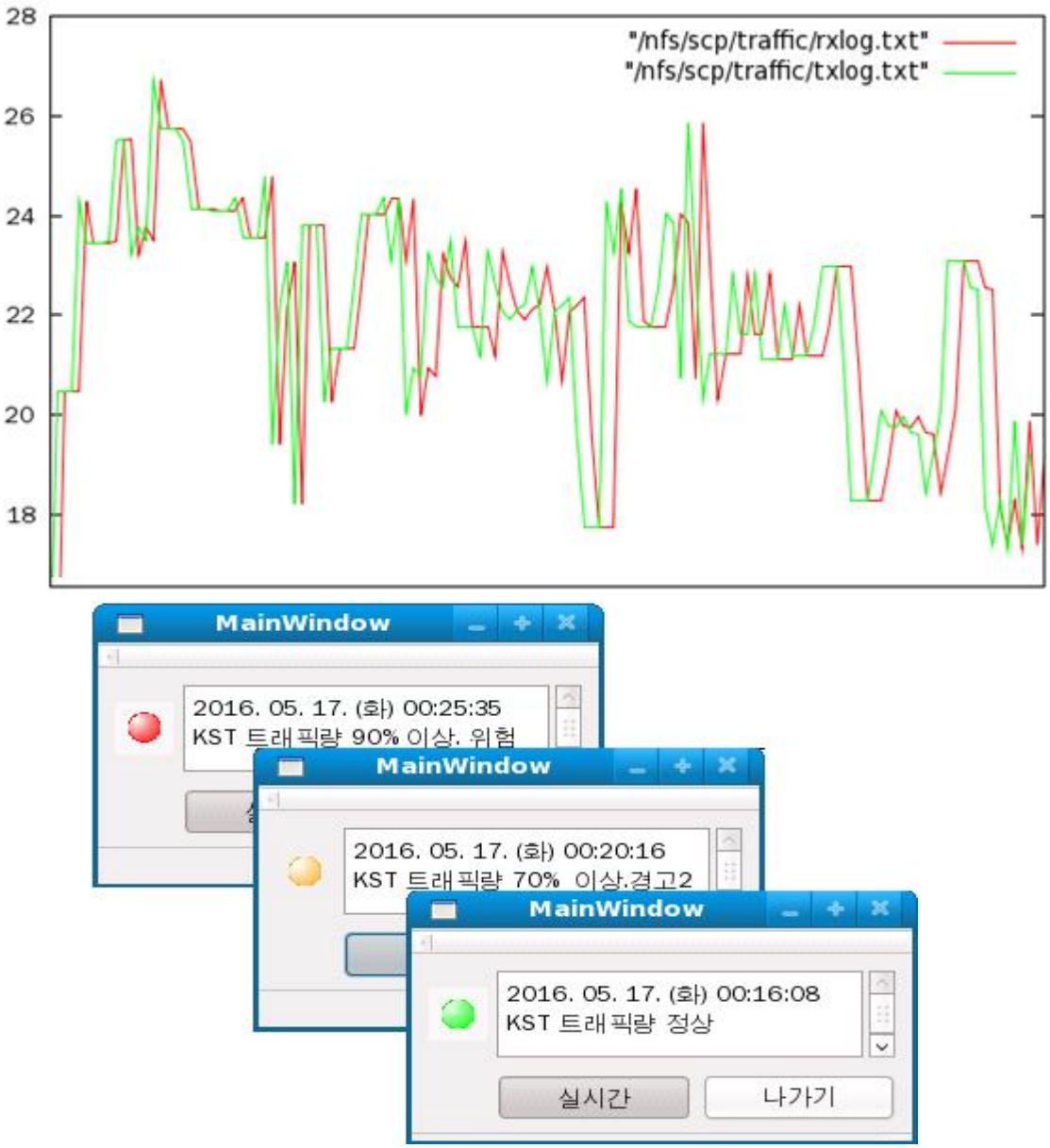


<그림 3 – 11 : 무결성 검사와 파일 변경 내용 >

3.2.6 트래픽 실시간 그래프 & 알림

트래픽 그래프의 경우 GunPlot를 이용하여 제작하였다.
데몬과 같이 백그라운드에서 작동되며 실시간으로 그래프를 관리자에게 보여 줄 수 있다. vnstat 와 같은 기존의 트래픽 분석 툴이 터미널 상에서만 나타내주며 지난 기록을 실시간으로 확인 할 수 없다는 점을 보안하기 위해 그래프로 나타내었다.

트래픽 량이 정해진 기준 수치의 일정 %를 넘어서게 되면 관리자에게 알림을 띄워준다.



< 그림 3 – 13 : 트래픽 실시간 그래프 와 사용량에 따른 알람 >

3.3 QT를 이용한 GUI 환경 구축

기존의 상용화 된 IDS 툴 과 이전 졸업 작품들은 리눅스 터미널 상에서만 메인 메뉴, 알림 등을 보여주어서 익숙하지 않은 사용자가 관리하기 불편하다는 문제점을 가지고 있다. 이러한 문제점을 해결하기 위해 C++ 기반의 QT 프로그래밍을 이용하여 GUI 환경을 구축함으로써, 관리자 점검 메뉴를 한눈에 관리 할 수 있으며, 서버에 문제점이 발견되었을 경우 기존의 터미널 경고 메시지가 아닌 팝업 형식의 알림 창을 띄워주어 실시간으로 서버의 문제점을 확인 할 수 있도록 하였다.

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include <QFile>
#include<QTextStream>
#include<QProcess>
#include<QDebug>
MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);
}

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::changeEvent(QEvent *e)
{
    QMainWindow::changeEvent(e);
    switch (e->type()) {
    case QEvent::LanguageChange:
        ui->retranslateUi(this);
        break;
    default:
```

< 그림 3 – 13 : QT를 이용한 GUI 환경 구축 >

3.4 사용 소스

arp.sh 소스코드

```
#!/bin/bash
while [ : ]
do
ip="`cat /ss/ip/ip.txt`"
B="`route | grep default | cut -f10 -d " "`"
nmap -v "$B" >> /ss/arp/nmap.txt
#A="`cat /ss/arp/nmap.txt | grep -w MAC | cut -d " " -f 3 |
head -1`"
#B="`cat /ss/arp/nmap.txt | grep -w scan | cut -d " " -f 5 |
head -1`"
A="`nmap -v 10.100.114.1 | grep MAC | cut -f3 -d " "`"
echo "$A" >> /ss/arp/log.txt
if [ -z "`grep -w "$A" /ss/arp/list.txt`" ]
then
        echo
 "================================================"          >>
/ss/log/arplog.txt
        date >> /ss/log/arplog.txt
        echo
 "================================================"          >>
/ss/log/arplog.txt
        echo "$B     $A" >> /ss/log/arplog.txt
        echo
 "================================================"          >>
/ss/log/arplog.txt
echo "`date` " >> /ss/log/qtarplog.txt
echo "정상적인 IP 및 MAC 주소 ">> /ss/log/qtarplog.txt
echo "IP : `cat /ss/arp/list.txt | tail -1` MAC : `cat
/ss/arp/list.txt | head -1`" >> /ss/log/qtarplog.txt
echo " 공격당한 IP 및 MAC 주소 " >> /ss/log/qtarplog.txt
echo "IP : `cat /ss/arp/nmap.txt | grep Scanning | cut -f2 -d
```

```
" " | head -1` MAC : `cat /ss/arp/nmap.txt | grep MAC | cut
-f3 -d " " | head -1`" >> /ss/log/qtarplog.txt
echo "`date | awk '{print $5}' | cut -f1 -d ":" ` " >>
/ss/log/qtarplog2.txt
echo "정상적인 IP 및 MAC 주소 ">> /ss/log/qtarplog2.txt
echo "IP : `cat /ss/arp/list.txt | tail -1` MAC : `cat
/ss/arp/list.txt | head -1`" >> /ss/log/qtarplog2.txt
echo " 공격당한 IP 및 MAC 주소 " >> /ss/log/qtarplog2.txt
echo "IP : `cat /ss/arp/nmap.txt | grep Scanning | cut -f2 -d
" " | head -1` MAC : `cat /ss/arp/nmap.txt | grep MAC | cut
-f3 -d " " | head -1`" >> /ss/log/qtarplog2.txt
sh /ss/arp/arpqt.sh
 fi
cat /ss/log/tmp.txt > /ss/arp/nmap.txt
done
```

port.sh 소스코드

```
#!/bin/bash
while [ : ]; do
DATE=`date`
iptables -L | grep ACCEPT | grep -v Chain | grep -v all | grep -v
icmp > /ss/port/port.txt
LIST=`cat /ss/port/port.txt | wc -l`
i=1
while [ $i -le $LIST ]; do
LISTA=`cat /ss/port/port.txt | cut -d ":" -f2 | head -$i | tail -1`
LISTB=`cat /ss/port/port.txt | awk '{print $2}' | head -$i | tail -1`
if [ -z "`cat /ss/port/list.txt | grep $LISTA | grep $LISTB`" ]; then
        echo "$DATE$LISTB$LISTA" >> /ss/port/log.txt
fi
let "i += 1"
done
```

```
sleep 1
done
```

alarm2.sh 소스코드 (port 알람)

```
#!/bin/bash
while [ : ]; do
DATE=`date`
LISTA=`cat /ss/port/log.txt | sort -u`
sleep 0.5
LISTB=`cat /ss/port/log.txt | sort -u`
COUNT=`cat /ss/port/log.txt | sort -u | wc -l`
echo $COUNT
if [ "$LISTA" != "$LISTB" ]; then
i=1
while [ $i -le $COUNT ]; do
if [ -n "`cat /ss/port/log.txt`" ]; then
LISTD=`cat /ss/port/log.txt | sort -u | head -$i | tail -1`
cat /ss/port/log.txt | grep $LISTD | head -1 | tail -1 >>
/ss/port/port.txt
else
sh /ss/port/alarm2.sh
fi
let "i += 1"
done
COUNT2=`cat /ss/port/log.txt | sort -u | wc -l`
if [ -n "`cat /ss/port/log.txt`" ]; then
LISTC=`cat /ss/port/log.txt | sort -u`
clear
echo "$DATE
----------------
비인가 포트 목록
```

```
----------------
$LISTC
$COUNT2 개의 비인가 포트가 열려있음" > /ss/port/alarmlog2.txt
echo "$DATE
----------------
비인가 포트 목록
----------------
$LISTC
$COUNT2 개의 비인가 포트가 열려있음" >> /ss/port/total.txt
sh /ss/port/qtstart.sh
fi
fi
done
```

service.sh 소스코드

```
#!/bin/bash
while [ : ]; do
`lsof | grep LISTEN | awk '{print $1}' | sort -u > /ss/service/service.txt`
COUNT=`cat /ss/service/service.txt | wc -l`
i=1
while [ $i -le $COUNT ]; do
LIST=`cat /ss/service/service.txt | head -$i | tail -1`
if [ -z "`cat /ss/service/list.txt | grep $LIST`" ]; then
        echo "$LIST" >> /ss/service/log.txt
fi
let "i += 1"
done
sleep 1
done
```

service2.sh 소스코드

```
#!/bin/bash
DATE=`date`
while [ : ]; do
LISTA=`cat /ss/service/log.txt | awk '{print $1}' | sort -u`
sleep 0.5
LISTB=`cat /ss/service/log.txt | awk '{print $1}' | sort -u`
COUNT=`cat /ss/service/log.txt | awk '{print $1}' | sort -u | wc -l`
alarm()
{i=1
while [ $i -le $COUNT ]; do
clear
echo "변화없음"
break
let "i += 1"
done
}ture()
{while [ : ]; do
DATE=`date`
COUNT2=`cat /ss/service/log.txt | sort -u | wc -l`
if [ -n "`cat /ss/service/log.txt`" ]; then
LISTC=`cat /ss/service/log.txt | sort -u`
clear
echo "$DATE
------------------
비인가 서비스 목록
------------------
$LISTC
$COUNT2 개의 비인가 서비스 실행 중"
echo -n "서비스를 종료하였습니까? (y) : "
read -t 10 answer
```

```
if [ "$answer" = "y" ]; then
echo "$DATE
------------------
비인가 서비스 목록
------------------
$LISTC
$COUNT2 개의 비인가 서비스 실행 중" >> /ss/service/alarmlog.txt
cat /ss/tmp > /ss/service/log.txt
cat /ss/tmp > /ss/service/service.txt
alarm
else
clear
continue
fi
fi
done
}if [ "$LISTA" != "$LISTB" ]; then
alarm
COUNT2=`cat /ss/service/log.txt | sort -u | wc -l`
if [ -n "`cat /ss/service/log.txt`" ]; then
LISTC=`cat /ss/service/log.txt | sort -u`
clear
echo "$DATE
------------------
비인가 서비스 목록
------------------
$LISTC
$COUNT2 개의 비인가 서비스 실행 중"
echo -n "서비스를 종료하였습니까? (y) : "
read -t 10 answer
if [ "$answer" = "y" ]; then
```

```
echo "$DATE
------------------
비인가 서비스 목록
------------------
$LISTC
$COUNT2 개의 비인가 서비스 실행 중" >> /ss/service/alarmlog.txt
cat /ss/tmp > /ss/service/log.txt
cat /ss/tmp > /ss/service/service.txt
alarm
else
clear
ture
fi
fi
fi
done
```

alarm.sh 소스코드 (service 알람)

```
#!/bin/bash
DATE=`date`
while [ : ]; do
LISTA=`cat /ss/service/log.txt | awk '{print $1}' | sort -u`
sleep 0.5
LISTB=`cat /ss/service/log.txt | awk '{print $1}' | sort -u`
COUNT=`cat /ss/service/log.txt | awk '{print $1}' | sort -u | wc -l`
alarm()
{i=1
while [ $i -le $COUNT ]; do
clear
echo "변화없음"
break
```

```
let "i += 1"
done
}ture()
{while [ : ]; do
DATE=`date`
COUNT2=`cat /ss/service/log.txt | sort -u | wc -l`
if [ -n "`cat /ss/service/log.txt`" ]; then
LISTC=`cat /ss/service/log.txt | sort -u`
clear
echo "$DATE
------------------
비인가 서비스 목록
------------------
$LISTC
$COUNT2 개의 비인가 서비스 실행 중"
echo -n "서비스를 종료하였습니까? (y) : "
read -t 10 answer
if [ "$answer" = "y" ]; then
echo "$DATE
------------------
비인가 서비스 목록
------------------
$LISTC
$COUNT2 개의 비인가 서비스 실행 중" >> /ss/service/alarmlog.txt
cat /ss/tmp > /ss/service/log.txt
cat /ss/tmp > /ss/service/service.txt
alarm
else
clear
continue
fi
```

```
fi
done
}if [ "$LISTA" != "$LISTB" ]; then
alarm
COUNT2=`cat /ss/service/log.txt | sort -u | wc -l`
if [ -n "`cat /ss/service/log.txt`" ]; then
LISTC=`cat /ss/service/log.txt | sort -u`
clear
echo "$DATE
------------------
비인가 서비스 목록
------------------
$LISTC
$COUNT2 개의 비인가 서비스 실행 중"
echo -n "서비스를 종료하였습니까? (y) : "
read -t 10 answer
if [ "$answer" = "y" ]; then
echo "$DATE
------------------
비인가 서비스 목록
------------------
$LISTC
$COUNT2 개의 비인가 서비스 실행 중" >> /ss/service/alarmlog.txt
cat /ss/tmp > /ss/service/log.txt
cat /ss/tmp > /ss/service/service.txt
alarm
else
clear
ture
fi
fi
```

```
fi
done
```

tcp.sh

```
#!bin/bash
ip="`ifconfig -a | grep "inet" | grep "Bcast:" | awk '{print $2}' |
awk -F: '{print $2}'`"

tcpdump src $ip > /ss/land/tcp.txt
```

land.sh

```
#!/bin/bash
ip="`ifconfig -a | grep "inet" | grep "Bcast:" | awk '{print $2}' |
awk -F: '{print $2}'`"

i=0
line=0
sec=0
sec2=0
while [ : ]; do

i=`expr $i + 1`

  n="`cat /ss/land/tcp.txt | wc -l`"

  if [ $i -le $n ]; then
    dst="`cat /ss/land/tcp.txt | head -$i | tail -1 | awk '{print $5}'
| awk -F. '{print $1"."$2"."$3"."$4}' | awk -F: '{print $1}'`"
    if [ "$dst" = "$ip" ]; then
```

```
      echo "`cat /ss/land/tcp.txt | head -$i | tail -1 | awk '{print $1}'`" >> /ss/land/land.txt

         if [ "`cat /ss/land/land.txt | wc -l`" -gt 0 ]; then
         sec="`cat /land/newland/land.txt | head -1 | tail -1 | awk -F: '{print $3}' | awk -F. '{print $1}'`"
echo "$sec"
         sec="`expr $sec - 1`"
         fi
      sec2="`cat /ss/land/land.txt | head -$i | tail -1 | awk -F: '{print $3}' | awk -F. '{print $1}'`"

      elif [ $sec2 -eq $sec ]; then
      cat /ss/log/tmp.txt > /ss/land/land.txt
fi

   else
i=0
   fi

done
```

alarm.sh (landattack)

```
#!/bin/bash
ip="`ifconfig -a | grep "inet" | grep "Bcast:" | awk '{print $2}' | awk -F: '{print $2}'`"

while [ : ]; do
```

```
line=`cat /ss/land/land.txt | wc -l`
if [ $line -gt 0 ]; then

time=`head -1 /ss/land/land.txt | awk -F. '{print $1}'`
fi
num=0

if [ $line -ge 30 ]; then
        echo -n "`date`
$time 랜드어택이 탐지되었습니다.
" > /ss/land/qtland.txt
                `cat /ss/land/save.txt > /etc/sysconfig/iptables-config`
                `iptables -A INPUT -s $ip -d $ip -j DROP`
                service iptables save
                service iptables start

                num=`cat /ss/land/land.txt | wc -l`
                echo "`date`
$time 랜드어택 공격 $num 회
" >> /ss/land/log.txt

                `cat /ss/land/tmp.txt > /ss/land/tcp.txt`
                `cat /ss/land/tmp.txt > /ss/land/land.txt`
        else
                continue

fi
done
```

login.sh 소스코드

```
#!/bin/bash
while [ : ]; do
IP="`who | grep -v "0.0)" | grep -v tty1 | awk '{print $5}' | sed
's/(//' | sed 's/)//'`"
sleep 1
IP2="`who | grep -v "0.0)" | grep -v tty1 | awk '{print $5}' | sed
's/(//' | sed 's/)//'`"
if [ -z "$IP" ]; then
cat /ss/log/tmp.txt > /ss/login/alarm.txt
fi
if [ "$IP" != "$IP2" ]; then
if [ -n "$IP2" ]; then
i=1
COUNT="`who | grep -v "0.0)" | grep -v tty1 | wc -l`"
cat /ss/log/tmp.txt > /ss/login/alarm.txt
while [ $i -le $COUNT ]; do
IP3="`who | grep -v "0.0)" | grep -v tty1 | awk '{print $5}' | sed
's/(//' | sed 's/)//' | head -$i | tail -1`"
echo "$IP3 에서 원격 접속 중 " >> /ss/login/alarm.txt
let "i += 1"
done
if [ -s /ss/login/alarm.txt ]
then
sh /ss/login/qtloginalarm.sh
fi
fi
fi
done
```

loginalarm.sh 소스코드

```
#!/bin/bash
cat /ahn/over/tmp.txt > /ahn/backdoor/WARNING.txt
cat /ahn/over/tmp.txt > /ahn/backdoor/DEL.txt
cat /ahn/over/tmp.txt > /ahn/backdoor/ADD.txt
cd /usr/local/fcheck
./fcheck -a > /ahn/backdoor/log.txt
COUNTA=`cat /ahn/backdoor/log.txt | wc -l`
War=`cat /ahn/backdoor/log.txt | grep -w "WARNING" | awk
'{print $3}'`
if [ -n "$War" ]; then
echo "$War" > /ahn/
backdoor/WARNING.txt
fi
Add=`cat /ahn/backdoor/log.txt | grep -w "ADDITION" | awk
'{print $3}'`
if [ -n "$Add" ]; then
echo "$Add" > /ahn/backdoor/ADD.txt
fi
Del=`cat /ahn/backdoor/log.txt | grep -w "DELETION" | awk '{print
$3}'`
if [ -n "$Del" ]; then
echo "$Del" > /ahn/backdoor/DEL.txt
fi
```

fcheck.sh 소스코드

```
#!/bin/bash
while [ : ]
do
# Telnet 접속 실패 로그
```

```
cat /var/log/secure | grep -w "FAILED LOGIN 1 FROM" | awk '{print $1,$2,$3,$10,$12}' | sed 's/.$//' >> /ss/login/log/log.txt
# SSH 접속 실패 로그
cat /var/log/secure | grep "["[sshd0-9]"]: Failed password" | awk '{print $1,$2,$3,$11,$9}' >> /ss/login/log/log.txt
# FTP 접속 실패 로그
cat /var/log/secure | grep "(vsftpd:auth): authentication failure" | awk '{print $1,$2,$3,$14,$13}' | sed 's/ruser=//' | sed 's/rhost=//' >> /ss/login/log/log.txt
cat /ss/login/log/log.txt | awk '{print $4}' | grep -v :: | sort -u > /ss/login/log/ip.txt
cat /ss/login/log/log.txt | awk '{print $5}' | sort -u > /ss/login/log/id.txt
cat /ss/log/tmp.txt > /ss/login/loginlog.txt
cat /ss/log/tmp.txt > /ss/login/log/ipcount.txt
cat /ss/log/tmp.txt > /ss/login/log/idcount.txt
cat /ss/log/tmp.txt > /ss/login/log/idban.txt
cat /ss/log/tmp.txt > /ss/login/log/ipban.txt
DATE=`date`
echo "==========================================
$DATE
==========================================" >> /ss/login/loginlog.txt
# ID 차단
idcount=`cat /ss/login/log/id.txt | wc -l`
idcount=`expr $idcount + 1`
echo "==========================================
ID 차단 List
==========================================" >> /ss/login/log/idban.txt
i=1
```

```
while [ $i -lt $idcount ]; do
idA=`cat /ss/login/log/id.txt | head -$i | tail -1`
idB=`cat /ss/login/log/log.txt | grep -w "$idA" | sort -u | wc -l`
if [ "$idA" = "`grep -w $idA /etc/passwd | cut -d ":" -f1 | head
-1`" ]; then
if [ $idB -ge 3 ]; then
        #passwd -l $idA > /dev/null
        echo "사용자 $idA 이(가) 비밀번호를 3회 이상 틀려 사용을
제한 합니다." >> /ss/login/qtlogin.txt
        echo "ID : $idA 틀린 횟수 : $idB" >>
/ss/login/loginlog.txt
        echo "₩"$idA₩", $idB" >> /ss/login/log/idcount.txt
        echo " $i  : $idA" >> /ss/login/log/idban.txt
fi
fi
let "i += 1"
done
#IP차단
ipcount=`cat /ss/login/log/ip.txt | wc -l`
ipcount=`expr $ipcount + 500`
echo "==========================================
IP 차단 List
==========================================" >>
/ss/login/log/ipban.txt
j=500
while [ $j -lt $ipcount ]; do
ipA=`cat /ss/login/log/ip.txt | head -$j | tail -1`
ipB=`cat /ss/login/log/log.txt | grep -w "$ipA" | sort -u | wc -l`
if [ $ipB -ge 3 ]; then
        #route add -host $ipA reject > /dev/null
        #echo "아이피 주소 $ipA 이(가) 비밀번호를 10회 이상 틀려
```

```
IP를 제한 합니다."
        echo "IP : $ipA 틀린 횟수 : $ipB" >> /ss/login/loginlog.txt
        echo "₩"$ipC₩", $ipB" >> /ss/login/log/ipcount.txt
        echo " $j : $ipA" >> /ss/login/log/ipban.txt
fi
let "j += 1"
done
if [ -z "`grep User /ss/login/log/idcount.txt`" ]; then
perl -p -i -e '$.==1 and print "User,Count₩n"' /ss/login/log/idcount.txt
fi
if [ -z "`grep IP /ss/login/log/ipcount.txt`" ]; then
perl -p -i -e '$.==1 and print "IP,Count₩n"' /ss/login/log/ipcount.txt
fi
cat /ss/login/log/idcount.txt > /ss/login/log/idcount.csv
done
```

report.sh

```
#!/bin/bash

clear

COUNTA=`cat /ahn/backdoor/WARNING.txt | wc -l`
COUNTB=`cat /ahn/backdoor/ADD.txt | wc -l`
COUNTC=`cat /ahn/backdoor/DEL.txt | wc -l`

echo "1. 변경된 파일 개 수 : $COUNTA
2. 추가된 파일 개 수 : $COUNTB
```

```
3. 삭제된 파일 개 수 : $COUNTC "

echo -n "확인 할 파일 리스트 : "
read num

case $num in

1)
echo "
변경된 파일 리스트"

if [ -n "`cat /ahn/backdoor/WARNING.txt`" ]; then

cat /ahn/backdoor/WARNING.txt

fi
;;
2)
echo "
추가된 파일 리스트"
if [ -n "`cat /ahn/backdoor/ADD.txt`" ]; then

cat /ahn/backdoor/ADD.txt

fi
;;
3)
echo "
삭제된 파일 리스트"
if [ -n "`cat /ahn/backdoor/DEL.txt`" ]; then
```

```
cat /ahn/backdoor/DEL.txt

fi
;;
esac
```

over.sh (시스템 전체 검사)

```
#!/bin/bash

cat /ss/log/tmp.txt > /ss/over/func.txt

cd /

i=1
COUNT=`cat /ss/over/list.txt | wc -l`

while [ $i -le $COUNT ]; do

FUNC=`cat /ss/over/list.txt | head -$i | tail -1`

cat /ss/log/tmp.txt > /ss/over/func/$FUNC.txt

`ack -a "$FUNC" ./* | grep -v Binary | cut -d ":" -f1 | sort -u >>
/ss/over/func/$FUNC.txt`

if [ -z "`cat /ss/over/func/$FUNC.txt`" ]; then
echo "이상없음" > /dev/null
else
echo "==========$FUNC==========" >> /ss/over/func.txt
cat /ss/over/func/$FUNC.txt >> /ss/over/func.txt
fi
```

```
let "i += 1"
done

#cat /ss/over/func.txt | sort -u > /ss/over/func.txt



date
```

over2.sh (변경파일만 검사)

```
#!/bin/bash

cat /ss/over/tmp.txt > /ss/over/funcm.txt
cat /ss/over/tmp.txt > /ss/over/TOTAL.txt

cat /ss/backdoor/WARNING.txt >> /ss/over/TOTAL.txt
cat /ss/backdoor/ADD.txt >> /ss/over/TOTAL.txt



COUNT=`cat /ss/over/list.txt | wc -l`

i=1

while [ $i -le $COUNT ]; do

FUNC=`cat /ss/over/list.txt | head -$i | tail -1`

cat /ss/log/tmp.txt > /ss/over/funcm/$FUNC.txt

j=1

COUNTB=`cat /ss/over/TOTAL.txt | wc -l`
```

```
while [ $j -le $COUNTB ]; do

route=`cat /ss/over/TOTAL.txt | head -$j | tail -1`

if [ -n "`ack -a "$FUNC" "$route" | grep -v Binary | cut -d ":" -f1
| sort -u`" ]; then

echo "$route" >> /ss/over/funcm/$FUNC.txt

fi

let "j += 1"

done

if [ -z "`cat /ss/over/funcm/$FUNC.txt`" ]; then
echo "이상없음" >> /dev/null
else
echo "==========$FUNC==========" >> /ss/over/funcm.txt
cat /ss/over/funcm/$FUNC.txt >> /ss/over/funcm.txt
fi

let "i += 1"
done
```

func.sh

```
#!/bin/bash

i=1

COUNT=`cat /ss/over/func.txt | wc -l`
```

```
while [ $i -le $COUNT ]; do

func=`cat /ss/over/func.txt | head -$i | tail -1`

if [ -z "`cat /ss/over/func.txt | grep $func`" ]; then
        echo "$func" >> /ss/over/log.txt
fi
let "i += 1"
done
```

back.sh

```
#!/bin/bash

find / -perm 4755 -exec ls -l {} ₩; > /ss/backdoor/back.txt

before="`cat /ss/backdoor/list.txt | wc -l`"
after="`cat /ss/backdoor/back.txt | wc -l`"

if [ "$before" -eq "$after" ]; then

echo "이상 없습니다."

else

i=1

cat /ss/backdoor/back.txt | grep –v "합계" > cat /ss/backdoor/back.txt2

cat /ss/backdoor/back2.txt | grep –v "합계" > cat /ss/backdoor/back.txt
```

```
rm ss/ackdoor/back2.tx

while [ $i -le $after ]; do

afterA=`cat /ss/backdoor/back.txt | awk '{print $8 }' | head -$i |
tail -1`

time=`ls –l $afterA | awk '{print $7}'`

if [ -z "`cat /ss/backdoor/list.txt | grep -w $afterA`" ]; then
        echo "$time $afterA 가 생성되었습니다."
fi

let "i += 1"
done

fi
```

tp.sh

```
#!/bin/bash
echo "vnstat start" >> /ss/scp/start/start.log
while [ : ]
do
vnstat -l -i eth0 > /ss/scp/traffic/log.txt
done
```

tp2.sh

```
#!/bin/bash
```

```
echo "trlog start" >> /ss/scp/traffic/start.log

while [ : ];do
cat /ss/scp/traffic/log.txt | cut -d "k" -f1 | cut -d ":" -f2 | sed 's/
//g' | grep "[0-9]" | sed '/Monitoringeth/d' >>
/ss/scp/traffic/rxlog.txt
cat /ss/scp/traffic/log.txt | cut -d "k" -f1 | cut -d ":" -f2 | sed 's/
//g' | grep "[0-9]" | sed '/Monitoringeth/d' >>
/ss/scp/traffic/rxlog3.txt
cat /ss/scp/traffic/log.txt | cut -d "k" -f1 | cut -d ":" -f2 | sed 's/
//g' | grep "[0-9]" | sed '/Monitoringeth/d' >>
/ss/scp/traffic/txlog.txt

sleep 1
done
```

reset.sh

```
#!/bin/bash
echo "resetlog start" >> /ss/scp/start/start.log
while [ : ]; do
PS=`ps -ef | grep -w "vnstat -l" | awk '{print $2}' | head -1`

if [ "`cat /ss/scp/traffic/log.txt | wc -c`" -gt 206 ]; then
        sleep 1
        cat /ss/log/tmp.txt > /ss/scp/traffic/log.txt
      `kill -9 $PS`
fi
done
```

start.sh

```
#!/bin/bash

cd /ss/scp
sh /ss/scp/scptmp.sh &
sh /ss/scp/shift.sh &
sh /ss/scp/traffic/reset.sh &
sh /ss/scp/traffic/tp.sh &
sh /ss/scp/traffic/tp2.sh &
```

alarm.sh (트래픽)

```
#!/bin/bash

LIMIT=300

while [ : ]; do

DATE=`date`

count="`cat /nfs/traffic/rxlog3.txt | wc -l`"
sleep 0.4
count2="`cat /nfs/traffic/rxlog3.txt | wc -l`"

if [ $count2 = $count ]; then
echo "No change" > /dev/null
else
while [ $count -le $count2 ]; do
sleep 0.6
```

```
M=`expr $count + 1`
A="`cat /nfs/traffic/rxlog3.txt | head -$count | tail -1 | cut -d "." -f1`"
B="`cat /nfs/traffic/rxlog3.txt | head -$M | tail -1 | cut -d "." -f1`"

LIMITA=`expr $LIMIT ₩* 90 / 100`
LIMITB=`expr $LIMIT ₩* 70 / 100`
LIMITC=`expr $LIMIT ₩* 50 / 100`

if [ "$A" -gt $LIMITA ]; then
C="Danger"
elif [ "$A" -gt $LIMITB ]; then
C="Warning2"
elif [ "$A" -gt $LIMITC ]; then
C="Warning"
else
C="Normal"
fi

if [ "$B" -gt $LIMITA ]; then
D="Danger"
elif [ "$B" -gt $LIMITB ]; then
D="Warning2"
elif [ "$B" -gt $LIMITC ]; then
D="Warning"
else
D="Normal"
fi

if [ "$C" = "$D" ]; then
```

```
echo "변화 없음" > /dev/null
elif [ "$D" = "Danger" ]; then
echo "$DATE 트래픽량 90%  이상. 위험"
echo "$DATE 트래픽량 90%  이상. 위험" >> /ss/tr/alarmlog.txt
elif [ "$D" = "Warning2" ]; then
echo "$DATE 트래픽량 70%  이상. 경고2"
echo "$DATE 트래픽량 70%  이상. 경고2" >> /ss/tr/alarmlog.txt
elif [ "$D" = "Warning" ]; then
echo "$DATE 트래픽량 50% 이상. 경고"
echo "$DATE 트래픽량 50% 이상. 경고" >> /ss/tr/alarmlog.txt
else
echo "$DATE 트래픽량 정상"
echo "$DATE 트래픽량 정상" >> /ss/tr/alarmlog.txt
fi

if [ -z "`cat /ss/tr/alarmlog.txt`" ]; then
perl -p -i -e '$.==1 and print
"=========================================₩n
        트래픽 로그"' /ss/tr/alarmlog.txt
e c h o
"=========================================" >>
/ss/tr/alarmlog.txt
fi

let "count += 1"

done
fi
done
```

QT main

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include<QString>
#include<QProcess>
#include<QDebug>
#include<QPixmap>
#include<QGraphicsScene>
#include<QGraphicsView>
#include<QImage>
#include<QFile>
MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);
//login
    QString arp = "/ss/pslog/login.sh";
        bool arpexist = QFile::exists(arp);
        if(true==arpexist){
QString png_path = "/ss/menu/mainmenu/green2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView->setScene(scene);
scene->addPixmap(    buf);
            }
        else
        {
QString png_path = "/ss/menu/mainmenu/red2.png";
```

```
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView->setScene(scene);
scene->addPixmap(buf);
      }
         //fcheck
         QString fcheck = "/ss/pslog/fcheck.sh";
              bool fcheckexist = QFile::exists(fcheck);
              if(true==fcheckexist){
QString png_path  =  "/ss/menu/mainmenu/green2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(18,18);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_2->setScene(scene);
scene->addPixmap(buf);
}
              else
              {
QString png_path  =  "/ss/menu/mainmenu/red2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_2->setScene(scene);
scene->addPixmap(buf);
           }
              //overflow
              QString over = "/ss/pslog/over.sh";
```

```
                    bool overexist = QFile::exists(over);
                    if(true==overexist){
QString    png_path    =    "/ss/menu/mainmenu/green2.png
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_3->setScene(scene);
scene->addPixmap(buf);
                        }
                    else
                    {
QString png_path = "/ss/menu/mainmenu/red2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_3->setScene(scene);
scene->addPixmap(buf);
                }
    //backdoor
                    QString back = "/ss/pslog/back.sh";
                        bool backexist = QFile::exists(back);
                          if(true==backexist){
QString png_path = "/ss/menu/mainmenu/green2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
                ui->graphicsView_4->setScene(scene);
                          scene->addPixmap(buf);
```

```
                        }
                    else
                    {
        QString png_path = "/ss/menu/mainmenu/red2.png";
                  QImage png(png_path);
                  QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
                QGraphicsScene* scene = new QGraphicsScene;
                 ui->graphicsView_4->setScene(scene);
                              scene->addPixmap(buf);
                  }
    //land
                        QString land = "/ss/pslog/autousb.sh";
                         bool landexist = QFile::exists(land);
                         if(true==landexist){
QString png_path = "/ss/menu/mainmenu/green2.png";
                QImage png(png_path);
                 QPixmap buf = QPixmap::fromImage(png);
                         buf=buf.scaled(21,21);
  QGraphicsScene* scene = new QGraphicsScene;
                         ui->graphicsView_5->setScene(scene);
                       scene->addPixmap(buf);
                            }
                         else
                         {
QString png_path = "/ss/menu/mainmenu/red2.png";
             QImage png(png_path);
             QPixmap buf = QPixmap::fromImage(png);
              buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
              ui->graphicsView_5->setScene(scene);
```

```
            scene->addPixmap(buf);
        //port
            QString port = "/ss/pslog/port.sh";
            bool portexist = QFile::exists(port);
                if(true==portexist){
            QString              png_path                          =
"/ss/menu/mainmenu/green2.png";
            QImage png(png_path);
             QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(18,18);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_10->setScene(scene);
scene->addPixmap(buf);
                    }
                  else
                   {
QString png_path  =  "/ss/menu/mainmenu/red2.png";
        QImage png(png_path);
        QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_10->setScene(scene);
scene->addPixmap(buf);
                                                        }
                                  //service
                      QString service = "/ss/pslog/service.sh";
                       bool serviceexist = QFile::exists(service);
                       if(true==serviceexist){
QString png_path  =  "/ss/menu/mainmenu/green2.png";
              QImage png(png_path);
             QPixmap buf = QPixmap::fromImage(png);
```

```
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_11->setScene(scene);
scene->addPixmap(buf);
                        }
                         else
                        {
QString png_path  =  "/ss/menu/mainmenu/red2.png";
                QImage png(png_path);
                QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_11->setScene(scene);
scene->addPixmap(buf);
                                                        }
                            //arp
                      QString arp = "/ss/pslog/arp.sh";
                    bool arpexist = QFile::exists(arp);
                   if(true==arpexist){
QString png_path  =  "/ss/menu/mainmenu/green2.png";
          QImage png(png_path);
          QPixmap buf = QPixmap::fromImage(png);
           buf=buf.scaled(21,21);
           QGraphicsScene* scene = new QGraphicsScene;
          ui->graphicsView_14->setScene(scene);
          scene->addPixmap(     buf);
           }
          else
          {
          QString              png_path                          =
"/ss/menu/mainmenu/red2.png";
```

```
          QImage png(png_path);
         QPixmap buf = QPixmap::fromImage(png);
          buf=buf.scaled(21,21);
          QGraphicsScene* scene = new QGraphicsScene;
           ui->graphicsView_14->setScene(scene);
           scene->addPixmap(buf);
                                                                    }
            //graf
            QString graf = "/ss/pslog/tp.sh";
           bool grafexist = QFile::exists(graf);
           if(true==grafexist){
             QString              png_path                          =
"/ss/menu/mainmenu/green2.png";
             QImage png(png_path);
            QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
 ui->graphicsView_16->setScene(scene);

                        scene->addPixmap(     buf);
                          }
                          else
                           {
QString png_path  =  "/ss/menu/mainmenu/red2.png";
             QImage png(png_path);
             QPixmap buf = QPixmap::fromImage(png);
 buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_16->setScene(scene);
scene->addPixmap(buf);
//land
```

```
                QString land= "/ss/pslog/land.sh";
bool landexist = QFile::exists(land);
              if(true==landexist){
QString png_path = "/ss/menu/mainmenu/green2.png";
            QImage png(png_path);
            QPixmap buf = QPixmap::fromImage(png);
 buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
 ui->graphicsView_17->setScene(scene);
scene->addPixmap(     buf);
      }
else
 {
QString png_path = "/ss/menu/mainmenu/red2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_17->setScene(scene);
scene->addPixmap(buf);
      }
          //loginalarm
QString login= "/ss/pslog/loginalarm.sh";
 bool loginexist = QFile::exists(login);
if(true==loginexist){
QString png_path = "/ss/menu/mainmenu/green2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_12->setScene(scene);
```

```
scene->addPixmap(      buf);
}
else
 {
QString png_path  =  "/ss/menu/mainmenu/red2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_12->setScene(scene);
scene->addPixmap(buf);
}
}
}
MainWindow::~MainWindow()
{
     delete ui;
}
void MainWindow::changeEvent(QEvent *e)
{
     QMainWindow::changeEvent(e);
     switch (e->type()) {
     case QEvent::LanguageChange:
          ui->retranslateUi(this);
          break;
     default:
          break;
     }
}
void MainWindow::on_pushButton_3_clicked()
{
```

```
    QString png_path = "/ss/menu/mainmenu/green2.png";
                        QImage png(png_path);
                        QPixmap buf =
QPixmap::fromImage(png);
                        buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
 ui->graphicsView_2->setScene(scene);
scene->addPixmap(buf);
QString program = ("/ss/backdoor/fcheck.sh");
QStringList arguments;
QProcess *myProcess = new QProcess(this);
myProcess->start(program, arguments);
myProcess->waitForFinished();
QString strOut = myProcess->readAllStandardOutput();
qDebug() << strOut;
QString program2 = ("/ss/qt/fcheck/fcheck.sh");
QStringList arguments2;
QProcess *myProcess2 = new QProcess(this);
myProcess2->start(program2, arguments2);
myProcess2->waitForFinished();
QString strOut2 = myProcess2->readAllStandardOutput();
qDebug() << strOut2;
}
void MainWindow::on_pushButton_6_clicked()
{
    QString png_path = "/ss/menu/mainmenu/red2.png";
                        QImage png(png_path);
                        QPixmap buf =
QPixmap::fromImage(png);
                        buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
```

```
ui->graphicsView_2->setScene(scene);
                                scene->addPixmap(buf);
QString program = ("/ss/backdoor/stopfcheck.sh");
                                    QStringList arguments;
QProcess *myProcess = new QProcess(this);
myProcess->start(program, arguments);
myProcess->waitForFinished();
QString strOut = myProcess->readAllStandardOutput();
                                    qDebug() << strOut;
}
void MainWindow::on_pushButton_4_clicked()
{
    QString png_path = "/ss/menu/mainmenu/green2.png";
                            QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
                            buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_3->setScene(scene);
                                scene->addPixmap(buf);
QString program = ("/ss/over/qtover2.sh");
QStringList arguments;
QProcess *myProcess = new QProcess(this);
myProcess->start(program, arguments);
                                    myProcess->waitForFinished();
QString strOut = myProcess->readAllStandardOutput();
                                    qDebug() << strOut;
}
void MainWindow::on_pushButton_7_clicked()
{
    QString png_path = "/ss/menu/mainmenu/red2.png";
                            QImage png(png_path);
```

```
QPixmap buf =
QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_3->setScene(scene);
scene->addPixmap(buf);
QString program = ("/ss/over/stopover.sh");
QStringList arguments;
QProcess *myProcess = new QProcess(this);
myProcess->start(program, arguments);
myProcess->waitForFinished();
QString strOut = myProcess->readAllStandardOutput();
qDebug() << strOut;
}
void MainWindow::on_pushButton_5_clicked()
{
    QString png_path = "/ss/menu/mainmenu/green2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_4->setScene(scene);
scene->addPixmap(buf);
QString program = ("/ss/backdoor/backdoor.sh");
QStringList arguments;
QProcess *myProcess = new QProcess(this);
myProcess->start(program, arguments);
myProcess->waitForFinished();
QString strOut = myProcess->readAllStandardOutput();
qDebug() << strOut;
}
```

```
void MainWindow::on_pushButton_8_clicked()
{
    QString png_path = "/ss/menu/mainmenu/red2.png";
                        QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
                          buf=buf.scaled(21,21);
ui->graphicsView_4->setScene(scene);
                          scene->addPixmap(buf);
QString program = ("/ss/backdoor/stopbackdoor.sh");
                              QStringList arguments;
QProcess *myProcess = new QProcess(this);
myProcess->start(program, arguments);
myProcess->waitForFinished();
QString strOut = myProcess->readAllStandardOutput();
qDebug() << strOut;
}
void MainWindow::on_pushButton_9_clicked()
{
    QString png_path = "/ss/menu/mainmenu/green2.png";
                        QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
                        buf=buf.scaled(21,21);
                          QGraphicsScene* scene = new
QGraphicsScene;
ui->graphicsView_5->setScene(scene);
scene->addPixmap(buf);
}
void MainWindow::on_pushButton_10_clicked()
{
```

```
    QString png_path = "/ss/menu/mainmenu/red2.png";
                        QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
                        buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_5->setScene(scene);
scene->addPixmap(buf);
}
void MainWindow::on_pushButton_19_clicked()
{
    QString png_path = "/ss/menu/mainmenu/green2.png";
                        QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
                        buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_10->setScene(scene);
                          scene->addPixmap(buf);
QString program = ("/ss/port/startport.sh");
QStringList arguments;
QProcess *myProcess = new QProcess(this);
myProcess->start(program, arguments);
myProcess->waitForFinished();
QString strOut = myProcess->readAllStandardOutput();
                              qDebug() << strOut;
}
void MainWindow::on_pushButton_24_clicked()
{
    QString png_path = "/ss/menu/mainmenu/red2.png";
                        QImage png(png_path);
                        QPixmap          buf          =
QPixmap::fromImage(png);
```

```
                                buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_10->setScene(scene);
                                  scene->addPixmap(buf);
QString program = ("/ss/port/stopport.sh");
                                        QStringList arguments;
QProcess *myProcess = new QProcess(this);
myProcess->start(program, arguments);
                                        myProcess->waitForFinished();
QString strOut = myProcess->readAllStandardOutput();
                                        qDebug() << strOut;
}
void MainWindow::on_pushButton_20_clicked()
{
    QString png_path = "/ss/menu/mainmenu/green2.png";
                                QImage png(png_path);
                                QPixmap          buf          =
QPixmap::fromImage(png);
                                buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_11->setScene(scene);
                                  scene->addPixmap(buf);
QString program = ("/ss/service/qtstartservice.sh");
                                        QStringList arguments;
QProcess *myProcess = new QProcess(this);
myProcess->start(program, arguments);
myProcess->waitForFinished();
QString strOut = myProcess->readAllStandardOutput();
                                        qDebug() << strOut;
}
void MainWindow::on_pushButton_25_clicked()
```

```
{
    QString png_path = "/ss/menu/mainmenu/red2.png";
                        QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
                        buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_11->setScene(scene);
                          scene->addPixmap(buf);
                          QString program =
("/ss/service/stopservice.sh");
                               QStringList arguments;
QProcess *myProcess = new QProcess(this);
myProcess->start(program, arguments);
myProcess->waitForFinished();
QString strOut = myProcess->readAllStandardOutput();
                               qDebug() << strOut;
}
void MainWindow::on_pushButton_21_clicked()
{
    QString png_path = "/ss/menu/mainmenu/green2.png";
                        QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
QString program = ("/ss/login/loingalarm.sh");
                        buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_12->setScene(scene);
                          scene->addPixmap(buf);
QString program2 = ("/ss/login/loginalarm.sh");
                               QStringList arguments;
QProcess *myProcess = new QProcess(this);
myProcess->start(program2, arguments);
```

```
myProcess->waitForFinished();
QString strOut = myProcess->readAllStandardOutput();
qDebug() << strOut;
}
void MainWindow::on_pushButton_26_clicked()
{
    QString png_path = "/ss/menu/mainmenu/red2.png";
                        QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
                        buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_12->setScene(scene);
                          scene->addPixmap(buf);
QString program = ("/ss/login/stoploginalarm.sh");
                              QStringList arguments;
QProcess *myProcess = new QProcess(this);
myProcess->start(program, arguments);
myProcess->waitForFinished();
QString strOut = myProcess->readAllStandardOutput();
                              qDebug() << strOut;
}
void MainWindow::on_pushButton_29_clicked()
{
    QString program = ("/ss/login/qtlogin.sh");
        QStringList arguments;
        QProcess *myProcess = new QProcess(this);
        myProcess->start(program, arguments);
        myProcess->waitForFinished();
        QString strOut = myProcess->readAllStandardOutput();
        qDebug() << strOut;
}
```

```
void MainWindow::on_pushButton_28_clicked()
{
    //login
        QString arp = "/ss/pslog/login.sh";
            bool arpexist = QFile::exists(arp);
            if(true==arpexist){
QString png_path = "/ss/menu/mainmenu/green2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
                                        buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView->setScene(scene);
scene->addPixmap(    buf);
                }
            else
            {
QString png_path = "/ss/menu/mainmenu/red2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
                                        buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView->setScene(scene);
scene->addPixmap(buf);
        }
            //fcheck
            QString fcheck = "/ss/pslog/fcheck.sh";
                bool fcheckexist = QFile::exists(fcheck);
                if(true==fcheckexist){
QString png_path = "/ss/menu/mainmenu/green2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
```

```
buf=buf.scaled(18,18);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_2->setScene(scene);
scene->addPixmap(buf);
                        }
                    else
                    {
QString png_path = "/ss/menu/mainmenu/red2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_2->setScene(scene);
scene->addPixmap(buf);
                }
                    //overflow
                    QString over = "/ss/pslog/over.sh";
                        bool overexist = QFile::exists(over);
                        if(true==overexist){
QString png_path = "/ss/menu/mainmenu/green2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_3->setScene(scene);
scene->addPixmap(buf);
                            }
                        else
                        {
QString png_path = "/ss/menu/mainmenu/red2.png";
QImage png(png_path);
```

```
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_3->setScene(scene);
scene->addPixmap(buf);
                    }
          //backdoor
                        QString back = "/ss/pslog/back.sh";
                            bool backexist = QFile::exists(back);
                                if(true==backexist){
QString png_path  =  "/ss/menu/mainmenu/green2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_4->setScene(scene);
scene->addPixmap(buf);
                                }
                            else
                            {
QString png_path  =  "/ss/menu/mainmenu/red2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_4->setScene(scene);
scene->addPixmap(buf);
                        }
          //land
                            QString land = "/ss/pslog/autousb.sh";
                                bool landexist = QFile::exists(land);
```

```
                                    if(true==landexist){
QString png_path = "/ss/menu/mainmenu/green2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_5->setScene(scene);
scene->addPixmap(buf);
                                        }
                                    else
                                    {
QString png_path = "/ss/menu/mainmenu/red2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_5->setScene(scene);
scene->addPixmap(buf);
                                                                    //port
QString port = "/ss/pslog/port.sh";
bool portexist = QFile::exists(port);
if(true==portexist){
QString png_path = "/ss/menu/mainmenu/green2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(18,18);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_10->setScene(scene);
scene->addPixmap(buf);
}
else
```

```
{
QString png_path = "/ss/menu/mainmenu/red2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_10->setScene(scene);
scene->addPixmap(buf);
                                                                    }
//service
QString service = "/ss/pslog/service.sh";
bool serviceexist = QFile::exists(service);
if(true==serviceexist){
QString png_path = "/ss/menu/mainmenu/green2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_11->setScene(scene);
scene->addPixmap(buf);
}
else
{
QString png_path = "/ss/menu/mainmenu/red2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_11->setScene(scene);
scene->addPixmap(buf);
}
```

```
//arp
QString arp = "/ss/pslog/arp.sh";
bool arpexist = QFile::exists(arp);
if(true==arpexist){
QString png_path = "/ss/menu/mainmenu/green2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_14->setScene(scene);
scene->addPixmap(     buf);
}
else
{
QString png_path = "/ss/menu/mainmenu/red2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_14->setScene(scene);
scene->addPixmap(buf);
}
//graf
QString graf = "/ss/pslog/tp.sh";
bool grafexist = QFile::exists(graf);
if(true==grafexist){
QString png_path = "/ss/menu/mainmenu/green2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
```

```
ui->graphicsView_16->setScene(scene);
scene->addPixmap(      buf);
}
else
{
QString png_path  =  "/ss/menu/mainmenu/red2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_16->setScene(scene);
scene->addPixmap(buf);
}
//land
QString land= "/ss/pslog/land.sh";
bool landexist = QFile::exists(land);
if(true==landexist){
QString png_path  =  "/ss/menu/mainmenu/green2.png";
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_17->setScene(scene);
scene->addPixmap(      buf);
}
else
{
QString png_path  =  "/ss/menu/mainmenu/red2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
```

```
ui->graphicsView_17->setScene(scene);
scene->addPixmap(buf);
}
//loginalarm
QString login= "/ss/pslog/loginalarm.sh";
bool loginexist = QFile::exists(login);
if(true==loginexist){
QString png_path = "/ss/menu/mainmenu/green2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_12->setScene(scene);
scene->addPixmap(     buf);
}
else
{
QString png_path = "/ss/menu/mainmenu/red2.png";
QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(21,21);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView_12->setScene(scene);
scene->addPixmap(buf);
                                                    }
                                        }
                                }
void MainWindow::on_pushButton_30_clicked()
{
    QString program = ("/ss/land/report.sh");
        QStringList arguments;
```

```
            QProcess *myProcess = new QProcess(this);
            myProcess->start(program, arguments);
            myProcess->waitForFinished();
            QString strOut = myProcess->readAllStandardOutput();
            qDebug() << strOut;
}
void MainWindow::on_pushButton_31_clicked()
{
     QString program = ("/ss/qt/report2/report2.sh");
            QStringList arguments;
            QProcess *myProcess = new QProcess(this);
            myProcess->start(program, arguments);
            myProcess->waitForFinished();
            QString strOut = myProcess->readAllStandardOutput();
            qDebug() << strOut;
}
```

QT Traffic

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include<QString>
#include<QProcess>
#include<QDebug>
#include<QPixmap>
#include<QGraphicsScene>
#include<QGraphicsView>
#include<QImage>
#include<QFile>
#include <QMessageBox>
MainWindow::MainWindow(QWidget *parent) :
     QMainWindow(parent),
```

```
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);
}
MainWindow::~MainWindow()
{
    delete ui;
}
void MainWindow::changeEvent(QEvent *e)
{
    QMainWindow::changeEvent(e);
    switch (e->type()) {
    case QEvent::LanguageChange:
        ui->retranslateUi(this);
        break;
    default:
        break;
    }
}

void MainWindow::on_pushButton_clicked()
{
    QFile file("/ss/scp/alarmlog2.txt");
      if(!file.open(QIODevice::ReadOnly))
          QMessageBox::information(0,"info",file.errorString());
      QTextStream in(&file);
  ui->textBrowser->setText(in.readAll());
  QString Danger = "/ss/scp/traffic/Danger.txt";
    QString Warning= "/ss/scp/traffic/Warning.txt";
        QString Warning2= "/ss/scp/traffic/Warning2.txt";
      bool arpexist = QFile::exists(Danger);
```

```
 bool arpexist2 = QFile::exists(Warning);
        bool arpexist3 = QFile::exists(Warning2);
    if(true==arpexist){
        QString png_path = "/ss/menu/mainmenu/red2.png";
                        QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(31,31);
QGraphicsScene* scene = new QGraphicsScene;
                        ui->graphicsView->setScene(scene);
                        scene->addPixmap(    buf);
        }
    else if(true==arpexist2)
    {
        QString png_path = "/ss/menu/mainmenu/orange.png";
                        QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
buf=buf.scaled(31,31);
QGraphicsScene* scene = new QGraphicsScene;
ui->graphicsView->setScene(scene);
                        scene->addPixmap(buf);
                    }
    else if(true==arpexist3)
    {
        QString png_path = "/ss/menu/mainmenu/orange.png";
                        QImage png(png_path);
QPixmap buf = QPixmap::fromImage(png);
                        buf=buf.scaled(31,31);
QGraphicsScene* scene = new QGraphicsScene;
                        ui->graphicsView->setScene(scene);
                        scene->addPixmap(buf);
                    }
```

```
        else
        {
            QString  png_path   =   "/ss/menu/mainmenu/green2.png";
                                    QImage  png(png_path);
QPixmap  buf  =  QPixmap::fromImage(png);
                                    buf=buf.scaled(31,31);
QGraphicsScene*  scene  =  new  QGraphicsScene;
                                      ui->graphicsView->setScene(scene);
                                      scene->addPixmap(buf);
                                  }
  }
```

QT login

```
#include  "mainwindow.h"
#include  "ui_mainwindow.h"
#include  <QFile>
#include<QTextStream>
#include<QProcess>
#include<QDebug>
MainWindow::MainWindow(QWidget  *parent)  :
      QMainWindow(parent),
      ui(new  Ui::MainWindow)
{
      ui->setupUi(this);
}

MainWindow::~MainWindow()
{
      delete  ui;
}
```

```
void MainWindow::changeEvent(QEvent *e)
{
    QMainWindow::changeEvent(e);
    switch (e->type()) {
    case QEvent::LanguageChange:
        ui->retranslateUi(this);
        break;
    default:
        break;
    }
}

void MainWindow::on_pushButton_clicked()
{
        QString INSERT=ui->lineEdit->text();
        QString INSERT2=ui->lineEdit_2->text();
        QFile file("/ss/qt/login/login.txt");
      if (!file.open(QIODevice::WriteOnly | QIODevice::Text))
    return;
        QTextStream out(&file);
    out << INSERT;
    out << INSERT2;
    QString program = ("/ss/qt/login/qtlogin.sh");
        QStringList arguments;
        QProcess *myProcess = new QProcess(this);
        myProcess->start(program, arguments);
        myProcess->waitForFinished();
        QString strOut = myProcess->readAllStandardOutput();
        qDebug() << strOut;
}
```

```
void MainWindow::on_pushButton_2_clicked()
{
    QString program = ("/home/kus/qustion/qustion.sh");
        QStringList arguments;
        QProcess *myProcess = new QProcess(this);
        myProcess->start(program, arguments);
        myProcess->waitForFinished();
        QString strOut = myProcess->readAllStandardOutput();
        qDebug() << strOut;
}
```

4. 결론

침입 탐지이라는 것은 대부분이 해킹 공격 방법을 기반으로 보안 방법을 강구하는 만큼 직접 IDS를 제작하기 위해서는 보안 대책 뿐 만 아니라 해킹 기법에 대해 공부를 해야 하기 때문에 기존의 IDS 툴을 이용해보며 리눅스 시스템 보호에 대해 공부하는 것보다 더 효과적 이였다. 또한 기존 터미널 방식 위주의 리눅스 IDS 툴에서 벗어나 GUI를 이용하여 보다 이용하기 좋은 관리 환경을 구축하였다.

하지만 부족한 부분도 있다. 기존의 IDS 툴을 사용하지 않으며 IDS를 제작하였기 때문에 기존 툴에 비해 아주 다양한 각도에서의 침입탐지를 방어하지는 못하기 때문에 이 부분에 대해서는 지속적으로 보다 다양한 해킹 및 침입탐지 방법에 대한 연구와 이해가 필요하다.

5. 참고 자료

- 참고 문헌

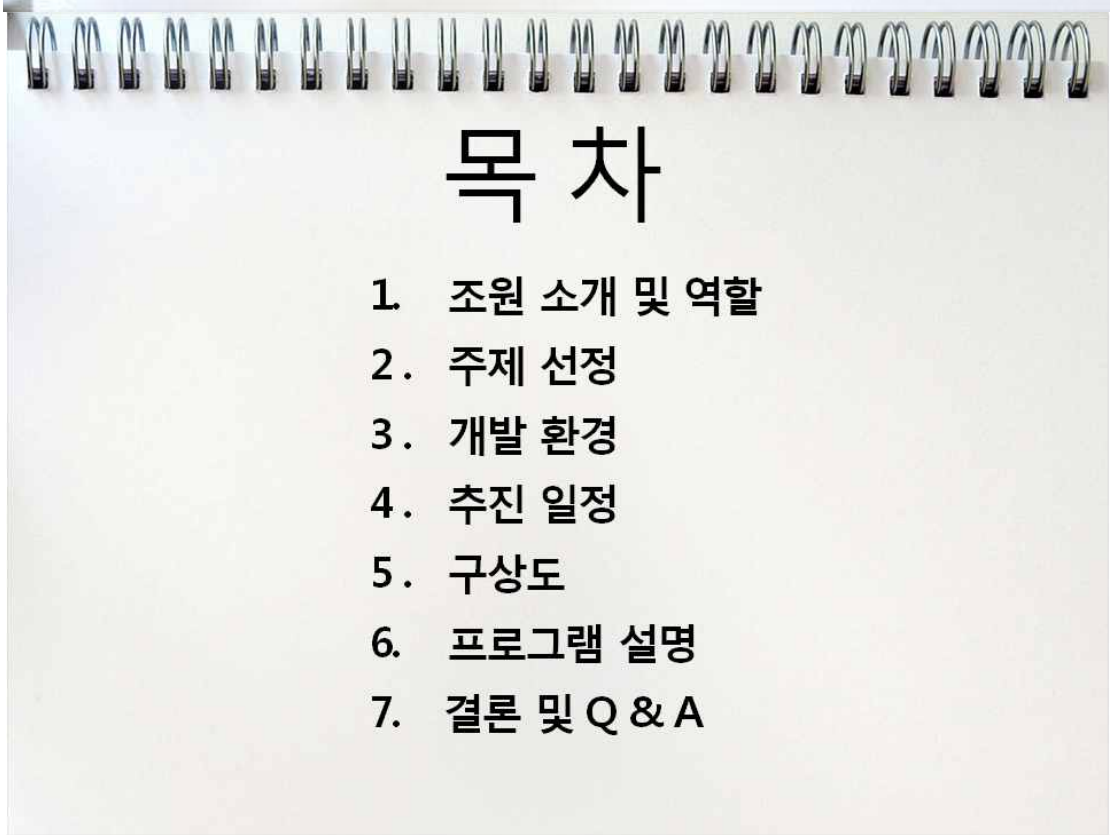
UNIX/Linux 시스템 관라자를 위한 쉘 스크립트 활용 가이드 (비팬북스)
리눅스를 활용한 회사 인프라 구축의 모든 것 (위키북스)
가장 쉬운 리눅스 시스템 관리 (비제이퍼블릭)
칼리 리눅스 입문자를 위한 메타스플로잇 중심의 모의 침투 (에이콘)
칼리 리눅스와 백트랙을 활용한 모의 해킹 (에이콘)
PHP프로그래밍 입문 (한빛 아카데미)
HTML5 CSS3 무작정 따라하기 (길벗)
QT4를 이용한 C++ GUI 프로그래밍 (ITC)

참고 사이트

QT Programming http://korone.net/
이재원님의 이글루 slog2.egloos.com
PHP 스쿨 http://www.phpschool.com
지성인놀이터 제타위키 http://zetawiki.com/wiki/Main_Page
HardCore in Programming http://kukuta.tistory.com/85
[리눅스] [해킹] Kali linux 2.0 을 사용한 해킹 정복
https://www.youtube.com/watch?v=I3TbPmhnZdY
Nmap을 이용한 네트워크 스캐닝과 방어하기
http://www.linuxlab.co.kr/docs/00-05-2.htm
[Linux] 서버 공격 방어를 위한 iptables 방화벽 설정 :: Karsei Rest Exhibition http://blog.karsei.pe.kr/6
통신 서비스 보안 http://webs.co.kr/?document_srl=433

6. 발표 자료

Server Sentry Tool 을
탑재한 IDS 구축
S.S (Server security)
담당교수님:양환석교수님
2016.05.25


목 차
1. 조원 소개 및 역할
2. 주제 선정
3. 개발 환경
4. 추진 일정
5. 구상도
6. 프로그램 설명
7. 결론 및 Q & A

조원 소개 및 역할

이 름	역 할
김의선	총괄 및 GUI 프로그래밍 ,쉘 프로그래밍,모의해킹
안재민	쉘 프로그래밍 , 모의해킹
황규남	웹 프로그래밍 , 자료조사
김미리	GUI 프로그래밍 , 웹 프로그래밍
손솔미	쉘 프로그래밍 ,모의해킹 ,웹 프로그래밍

주제 선정

기존 보안 툴의 취약점을 모의 해킹을 통해 알아보고 기존 사용법이 복잡한 터미널 위주의 보안 시스템의 틀에서 벗어나 사용이 편리한 그래픽 기반 인터페이스 방식 IDS를 제작 후 서버 구축을 하기 위해 주제를 선정

개발 환경

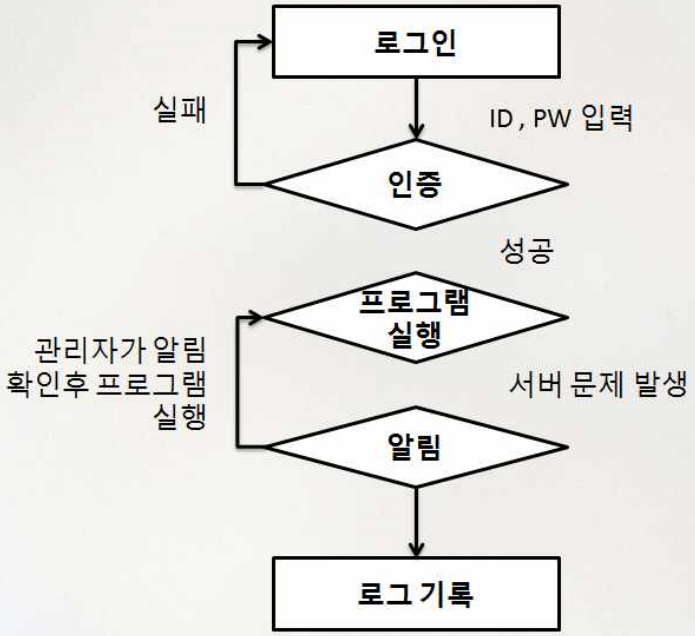
개발 환경 : Fedora 11 , Kali Linux , Window 7

개발 도구 : C언어, R언어, 쉘 스크립트 , QT4(C++), GNU Plot ,HTML , PHP , J query JavaScript , XML

추진일정

내 용	월 별 일 정 계 획										
	3월				4월				5월		
	1주	2주	3주	4주	1주	2주	3주	4주	1주	2주	3주
침 입 탐 지											
로 그 분 석											
모 의 해 킹											
대 응 및 보 안 대 책											
웹 페 이 지 구 축											
최 종 작 업											

구상도



프로그램 설명

(1) 메인 화면

로그인후

프로그램 설명

(2) Arp 스푸핑

스푸핑(Spoofing) 이란?

스푸핑의 사전적 뜻은 [속인다] 라는 의미로

ARP 스푸핑은 상대방의 MAC 주소를 속여 패킷을 가로채는 공격을 의미

ARP spoofing 공격전

```
Nmap scan report for 10.100.114.1
MAC Address: 10:0E:7E:2A:48:01
```

프로그램 설명

(2) Arp 스푸핑

```
HWaddr 60:eb:69:7e:0b:6b
```

```
root@kali:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 60:eb:69:7e:0b:6b
          inet addr:10.100.114.148  Bcast:10.100.114.255
```

Host List

IP Address	MAC Address	Description
10.100.114.1	10:0E:7E:2A:48:01	
10.100.114.44	14:DD:A9:54:D0:0B	
10.100.114.45	1C:39:47:B0:48:59	

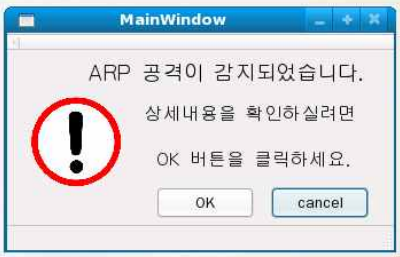
Delete Host | Add to Target 1 | Add to Tar

Unified sniffing was stopped.
Starting Unified sniffing...
DHCP: [10.100.114.1] ACK : 10.100.114.82 255.255.255.0 GW 10.100.114.1 DNS 114.71.195.130
DHCP: [10.100.114.1] ACK : 10.100.114.241 255.255.255.0 GW 10.100.114.1 DNS 114.71.195.130

```
Nmap scan report for 10.100.114.1
MAC Address: 10:0E:7E:2A:48:01
```

↓

```
Nmap scan report for 10.100.114.1
MAC Address: 60:EB:69:7E:0B:6B
```



프로그램 설명

(3) Land Attack

공격자 **피해자**

정상적인 통신일 경우 메시지에 대한 응답이 정상적으로 이루어진다 Land Attack이 실행되면 송 · 수신지의 IP주소가 피해자의 IP주소로 설정되며 , 피해자는 답신을 자기자신에게 보내게 됨으로 해당 패킷의 무한루프를 유도하게 된다

프로그램 설명

(3) Land Attack

```
[root@localhost land]# ping 10.100.114.55
PING 10.100.114.55 (10.100.114.55) 56(84) bytes of data.
64 bytes from 10.100.114.55: icmp_seq=1 ttl=64 time=0.037 ms
64 bytes from 10.100.114.55: icmp_seq=2 ttl=64 time=0.049 ms
64 bytes from 10.100.114.55: icmp_seq=3 ttl=64 time=0.039 ms
64 bytes from 10.100.114.55: icmp_seq=4 ttl=64 time=0.041 ms
```

```
--- 10.100.114.55 ping statistics ---
21 packets transmitted, 21 received, 0% packet loss, time 20713ms
rtt min/avg/max/mdev = 0.037/0.040/0.049/0.007 ms
```

공격전

프로그램 설명

(3) Land Attack

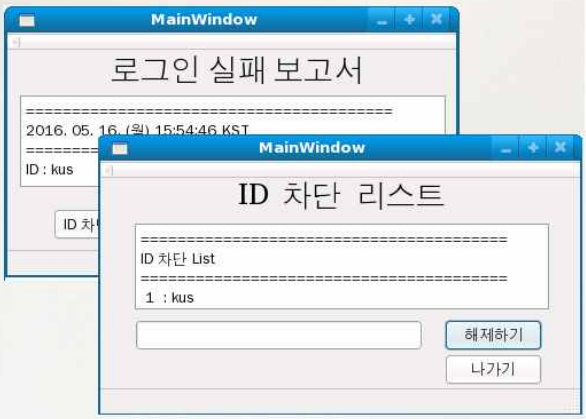
```
02:11:28.312048 IP 10.100.114.140.52284 > 65.55.2.82.http: . ack 8681 win 254
02:11:28.749448 IP 10.100.114.55.robcad-lm > 10.100.114.55.81: S 1739568602:1739568
602(0) win 512
02:11:29.749522 IP 10.100.114.55.mvx-lm > 10.100.114.55.81: S 703112770:703112770(0
) win 512
02:11:30.699093 IP 10.100.114.140.52284 > 65.55.2.82.http: P 12959:13168(209) ack 8
681 win 254
```

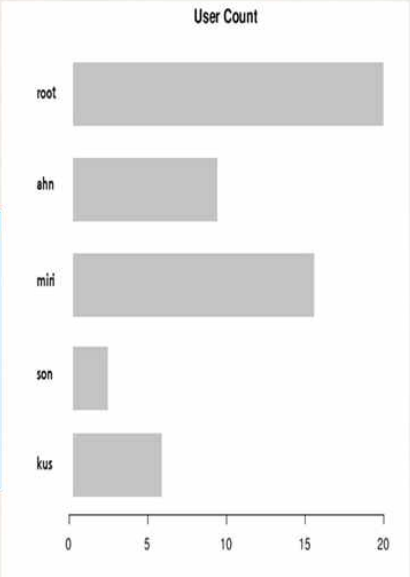
공격후

```
[root@localhost ~]# ping 10.100.114.55
PING 10.100.114.55 (10.100.114.55) 56(84) bytes of data.
--- 10.100.114.55 ping statistics ---
12 packets transmitted, 0 received, 100% packet loss, time 11119ms
```

프로그램 설명

(4) 크래킹 탐지





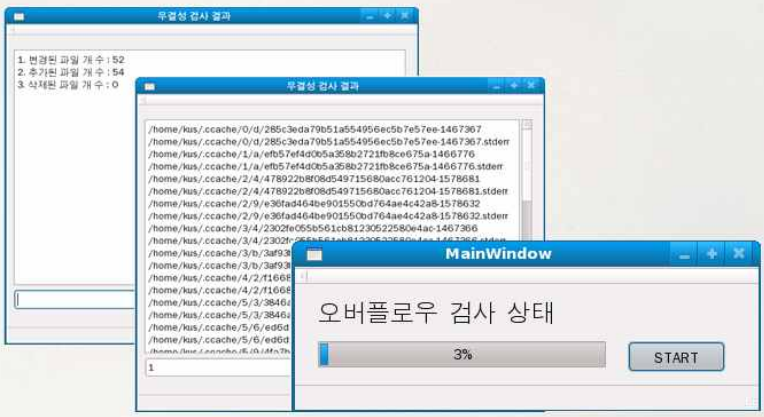
원격 접속 (Telnet, SSH 등) 으로 3회 이상 접속 실패 시 접속 기록을 남기며 해당 ID, IP 사용을 제한

프로그램 설명

(5)외부 접속 알림

프로그램 설명

(6) 무결성 검사 & 취약함수 감지

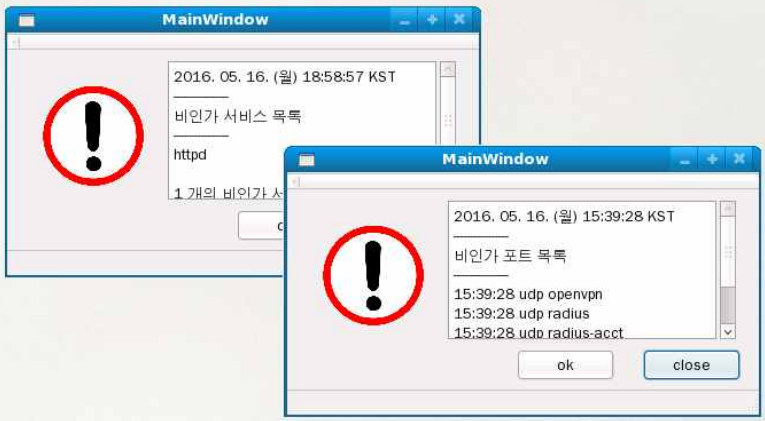


무결성 검사로 변형된 파일을 확인한 뒤

취약함수 검사를 통해 오버플로우 공격 을 탐지

프로그램 설명

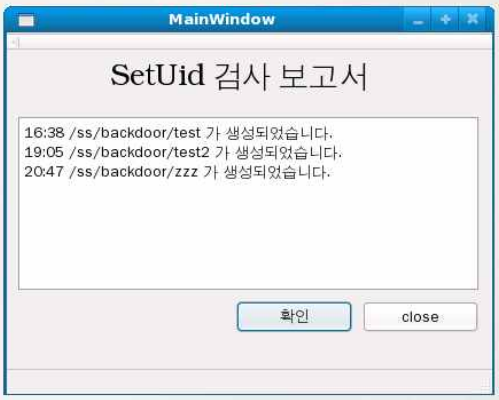
(8) 포트 & 서비스 확인



허가되지 않은 포트 및 서비스에 대한 관리자 알림

프로그램 설명

(7) 무결성 검사 & SetUid 검사

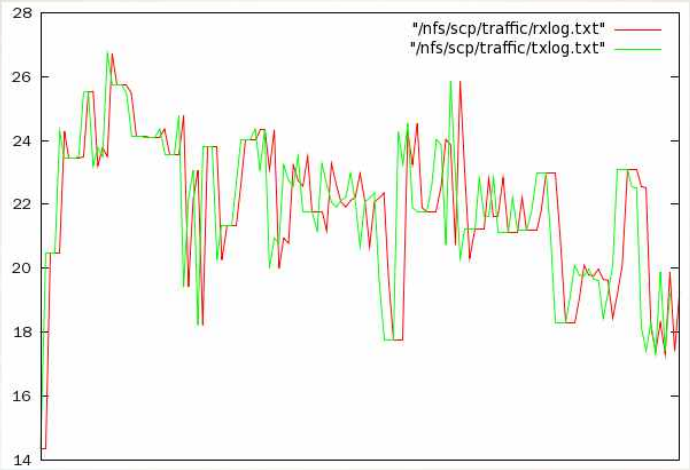


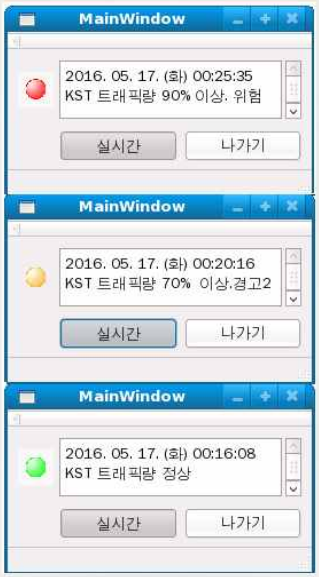
무결성 검사로 변형된 파일을 확인한 뒤

SetUID 검사를 통해 백도어 의심 공격 을 탐지

프로그램 설명

(9) 트래픽 실시간 그래프 & 알람





결론

시스템 개발 진행

- 해킹 공격 방법을 기반으로 한 IDS 직접 제작
- 기존의 터미널 방식을 벗어난 GUI 환경 구축

부족한 부분

- 다양한 각도의 침입탐지 불가
- 지속적인 해킹 및 침입탐지 방법 연구와 이해가 필요

Q & A

감사합니다